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

1.1 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 increased use of surveillance cameras a lot of research is done to automate information extraction from those cameras [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]. In Crowd Counting the amount of pedestrians present in the frame is counted. Where pedestrians in low density areas 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] using density maps.

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 the amount of required camera's increases quickly, this can become an 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, rather than having cameras in the whole shop.

This research area of Crowd Line Crossing is much less researched [Ma and Chan, 2013, Zhao et al., 2016, Zheng et al., 2019]. By estimation both a Flow Map and a Crowd Density Map (Crowd Counting) the flow of pedestrians can be obtained and the amount of people entering and exiting an area can be counted.

In early papers on Crowd Crossing Counting prediction was done using keypoint extraction and feeding the keypoints to 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], Flow Estimation [Sun et al., , Dosovitskiy et al., 2015] and Unsupervised Flow Estimation [Yu et al., 2016, Janai et al., 2018, Liu et al., 2019a, Liu et al.,]. Which sparked the research in those fields.

Reframe to new life to this field

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 lots 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.

1.2 Contributions

The latest Neural Networks for Crowd Crossing Counting [Zhao et al., 2016] learns both Crowd Counting and Flow Estimation in a supervised way. However the labelling process of both these methods are a time consuming method, especially labelling the images for Flow Estimation. To reduce the labour intensive task of labelling, a new method is introduced to use and train unsupervised Flow Estimation during Line Crossing Counting.

Additionally a multi-headed model is proposed which trains the Crowd Counting in a supervised matter and the Flow Estimation in an unsupervised matter. The model utilizes a shared encoder, while two distinct decoders are used to predict each an individual task.

To further utilize the availability of both the predictions a flow enhanced model is proposed. This model uses the Flow Estimation to further enhance the Crowd Counting prediction.

Due to dated datasets three new datasets are proposed, which are used for comparing the proposed networks against several strong existing baselines. Two datasets contain pedestrians, which are labeled with a self-build labeler. The third datasets contains cars on the road to show the generality of the system.

Lastly thorough research is done on the usability of the presented system in real world scenario's. Impact on video frame rate are tested and the processing time is compared.

1.3 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.
- **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.

Reframe ROI and LOI in the introduction or in the rest of the paper

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 counted pedestrians 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

With the introduction of Convolutional Neural Networks in the field of Crowd Counting density maps were proposed to count pedestrians [Zhang et al., 2016, Liu et al., 2019b, Li et al., 2018]. A *density map* (Figure 2.1) used for Region of

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

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 the location of the individual, 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).

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

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

Early networks for density estimation were mostly multi-column model [Zhang et al., 2016], which use different sizes of filters to detect pedestrians of different sizes. With CSR-Net [Li et al., 2018] dilation filters were introduced. Dilation filters (Figure **??**) enlarge the filter area without increasing the amount of parameters. With dilation filters the necessity of using multi-column models was gone.

Other papers focused on improving the performance using extra information [Shi et al., 2018, Shi et al., 2019, Liu et al., 2019b], such as global density [Shi et al., 2019] and variable Gaussians for density map generation [Zhang et al., 2016, Li et al., 2018, Wan and Chan, 2019]. A third one is adding motion by doing video-based counting.

Maybe add more papers??

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 lots 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].

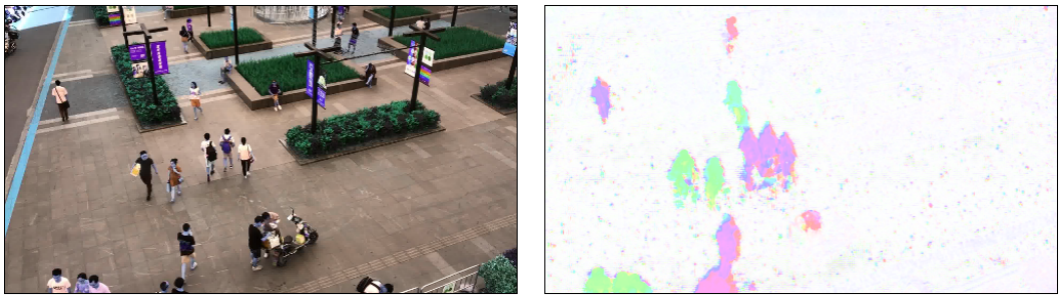


Figure 2.2: Example of generated velocity map on the right side, for the left image

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 models 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 there is a 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 these datasets and therefore perform rather poor on real world applications. One 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 (Top and bottom point in figure 2.3)

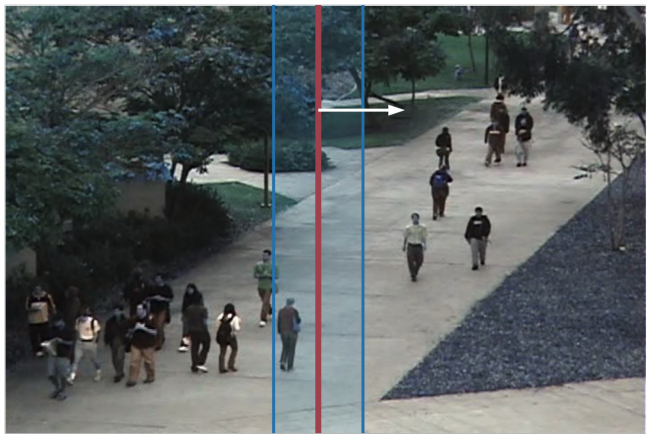


Figure 2.3: Example of LOI, red: LOI, blue: LOI area

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 (Blue area in figure 2.3), 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.

In [Zhao et al., 2016] both density map and velocity map are predicted by a unified network and merged together to obtain the line crossing counts. The unified model is based on FlowNetSimple [Dosovitskiy et al., 2015] with an extra output layer to predict the density map. The whole network is trained in a supervised manner. For the line crossing they use the next formulas, where formula 2.6 is the crossing for each video.

$$C_1^{(t)} = \sum_{p \in R_1} \mathbf{1}(v_1^{(t)}(p) \geq d(p, p_b)) \cdot \mathcal{D}^{(t)}(p), \tag{2.4}$$

$$C_2^{(t)} = \sum_{p \in R_2} \mathbf{1}(v_2^{(t)}(p) \geq d(p, p_b)) \cdot \mathcal{D}^{(t)}(p), \tag{2.5}$$

$$c_1 = \sum_{\{t|t\in T\}} C_{1,t}, \quad c_2 = \sum_{\{t|t\in T\}} C_{2,t} \tag{2.6}$$

Add extra information of the symbols

In [Zheng et al., 2019] a fast non-CNN based method is proposed based on a SVM, linear regression and the Lucas-Kanade optical flow tracker. This is much faster due to the reduction of complexity, but lacks the flexibility of a Neural Network and has difficulties with denser areas.

It uses the idea of [Zhao et al., 2016] to discard the slicing method and uses pairwise 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 2.7 a single velocity $v_{t,r}$ is calculated with a weighted average over all moving keypoints inside the region.

$$v_{t,r} = \frac{\sum_p w_r(p) \cdot v^{(t)}_{towards}(p)}{\sum_p w_r(p)} \tag{2.7}$$

Preliminary results show that the binning on a region-level is not improving the Neural Network models to increase accuracy. However by introducing smoothing in formula 2.6 performance increases, this is why in the method an updated line crossing formula is used.

Chapter 3

Method

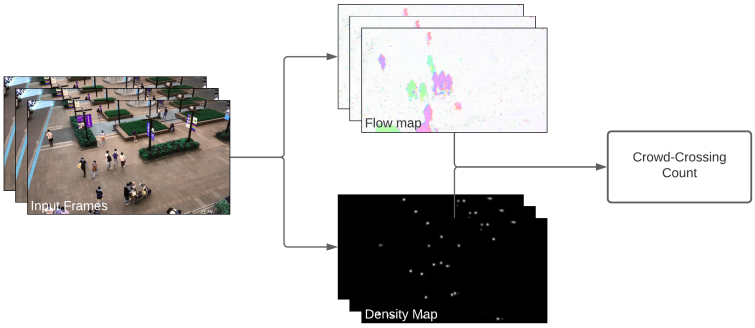


Figure 3.1: Overview of Crowd-Crossing Counting pipeline

In this chapter the proposed method for Crowd Crossing Counting is explained. As shown in figure 3.1, the full pipeline of prediction is based in two stages. First we predict both the density map and the velocity map, then we use these maps to predict the Crowd-Crossing Count. In section *Velocity Map and Density Map* are the methods explained to train/predict both the velocity map and density map. In section *LOI counting* the method is explained to predict the Crowd-Crossing Count based on those two intermediate maps.

3.1 Velocity Map and Density Map

Whereas earlier papers rely on velocity maps and density maps [Zhao et al., 2016], the amount of labelling required to train the models is high and creates an extra barrier for real world applications. To increase the usability of the method, an unsupervised velocity map estimation is used.

Training to predict both maps is being done in a paralel matter using a unified loss function described in equation 3.1. For the density loss, L_c, a traditional L2 loss is used (Figure 3.2). For the velocity loss L_v, the photometric loss from figure 3.3 [Yu et al., 2016, Janai et al., 2018] is used.

$$L_t = L_v + \lambda \cdot L_c \tag{3.1}$$

For training a pair of two consecutive frames is taken (I_1 and I_2) with a groundtruth

C, which is the ground truth density of the first frame. The model optimizes to output $\widetilde{C}$ and $\widetilde{V}$.

$$L_c = \sum (C - \widetilde{C})^2 \tag{3.2}$$

In figure 3.4, I_2^w and I_1^w are warped images with the predicted velocity $\widetilde{V}$. I_2^w is I_2 backward warped, so it optimizes to be equal to I_1 and I_1^w is I_1 forward warped to optimize to I_2. ψ is a robust loss function defined as in figure 3.4 with $\epsilon = 0.01$ and $q = 0.4$ (As used in [Liu et al., 2019a])

$$L_p = \sum \psi \left(I_1 - I_2^w\right) + \sum \psi \left(I_2 - I_1^w\right) \tag{3.3}$$

$$\psi(x) = (|x| + \epsilon)^q \tag{3.4}$$

As explained in the background chapter Velocity Map Prediction and Density Map prediction are two widely researched fields. Two separate models would be capable of predicting both the Velocity Map and Density Map accurately. However this produces extra overhead using two fully seperate models.

Therefore a new model is proposed which unifies both velocity map and density map prediction. Sharing a single encoder and two separate decoders for each task. Additionally several additions are made to the model to enhance the density map prediction using intermediate results of the velocity map decoder.

3.1.1 Models

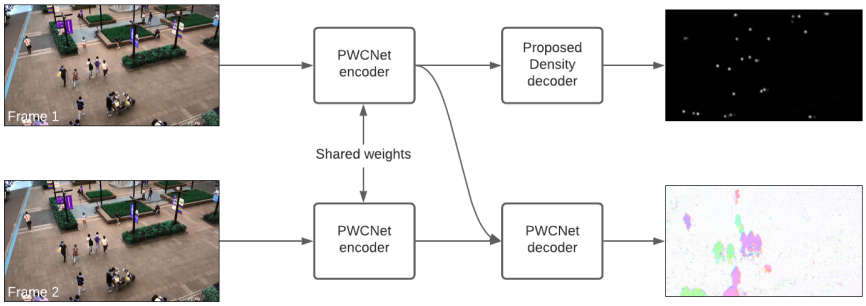


Figure 3.2: Unified model

The proposed model is a multi-headed model with a shared encoder shown as in figure 3.2 . The model uses the original PWCNet network [Sun et al.,] as backbone. The proposed model shares the encoder and decoder of the PWCNet, but adds a second decoder to predict a density map as well, as shown in figure 3.2.

PWCNet

The PWCNet network is a small two frame input flow estimator, which at the time of publication was the best performing supervised flow estimator and to the best of our knowledge still the best performing two frame model currently published. The network uses a u-shaped network of 6 different levels. Each level uses warping (correlation between warped image and first image) and a cost volume (Fig 3.3) to

refine the estimated flow estimation. Finally a context network is used to further refine the output.

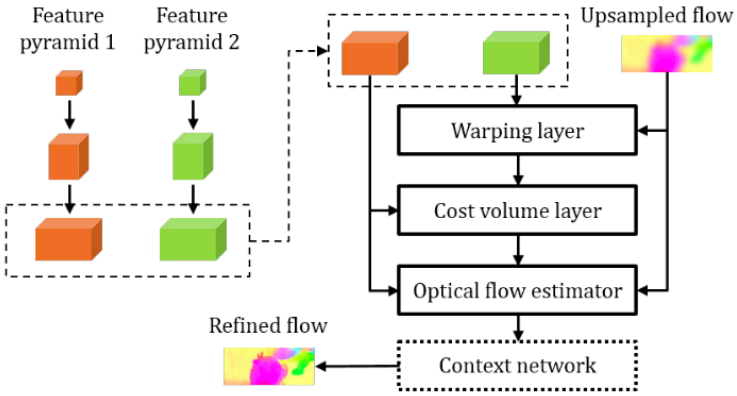


Figure 3.3: PWCNet architecture

Density map decoder

The density decoder contains of 5 upsampler modules with a final context network. Each upsampler module upsamples the output of the previous upsampler and concatenates it with the output of the encoder which are then smoothed with two regular conv-layers. The context module contains of 4 dilation layers with a respective dilation of 2,4,8,16 with a final layer to produce the density map.

Add figure to show the upsampler

Additional context

The baseline density map decoder predicts a density only based on the first frame of the pair (Fig 3.2). However to the velocity map prediction, a lot of extra information is at our disposal to further enhance the density map prediction (Fig 3.4). Several further enhancements will be tested in this thesis:

- Positional information, by adding using the encoder information of frame 2
- Flow information, by adding the raw cost volume of the flow estimator to the density decoder
- Warped information, by adding the warped positional information to the decoder

3.2 LOI counting

The second stage merges both the velocity map and density map into two line crossing numbers. For our approach the main idea of [Zhao et al., 2016] is taken (Explained in Related Work), however the current paper uses some simplifactions, by reframing the pixel-level counting in the following way. The approach is much more theoretical correct.

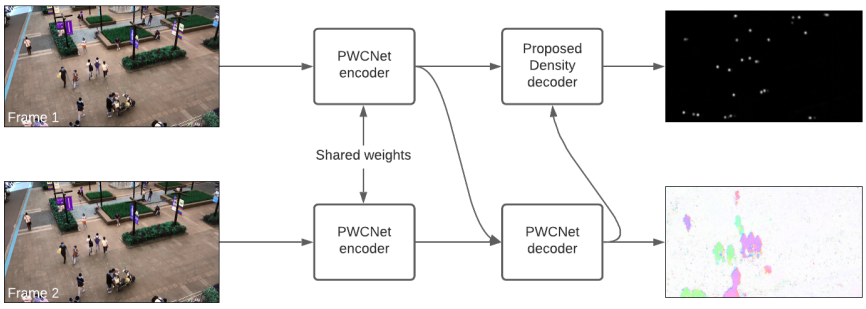


Figure 3.4: Flow enhanced model

We define v_{perp} (White arrow in figure 2.3) 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.

Add more visual description in same image as figure 2.3, explain that this assumption of left/right is without losing generality

The velocity towards the LOI is then defined as the dot-product of V_t and v_{perp} (Equation 3.5).

$$Q_t(p) = V_t(p) \cdot v_{perp} \tag{3.5}$$

$$\begin{aligned} c_{1,t} &= \sum_{\{p \in M_1 | Q_t(p) > 0\}} C_t(p) \cdot \frac{Q_t(p)}{d} \\ c_{2,t} &= \sum_{\{p \in M_2 | Q_t(p) < 0\}} C_t(p) \cdot \frac{-Q_t(p)}{d} \end{aligned} \tag{3.6}$$

Use two methods, smoothed one and non smoothed one (Zhao). Non-smoothed in background?

Then the LOI count on timestep t is defined in equation 3.6. 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.

3.2.1 Realigning

In [Zhao et al., 2016] a supervised method of aligning the velocity map and density map is used. However due to the unsupervised prediction of the velocity map in this thesis it is not possible to use this alignment. As shown in figure 3.5 this misalignment is an huge issue when multiplying the velocity map and density map in a pixel-wise matter. The blobs predicted in the density map are often much bigger than the flow of the pedestrian predicted. Additionally, for several Crowd datasets the tagging of the pedestrian is done on the head, which is why the blobs are ofter predicted over the head of the pedestrian.

To tackle this misalignment 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 3.5 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.

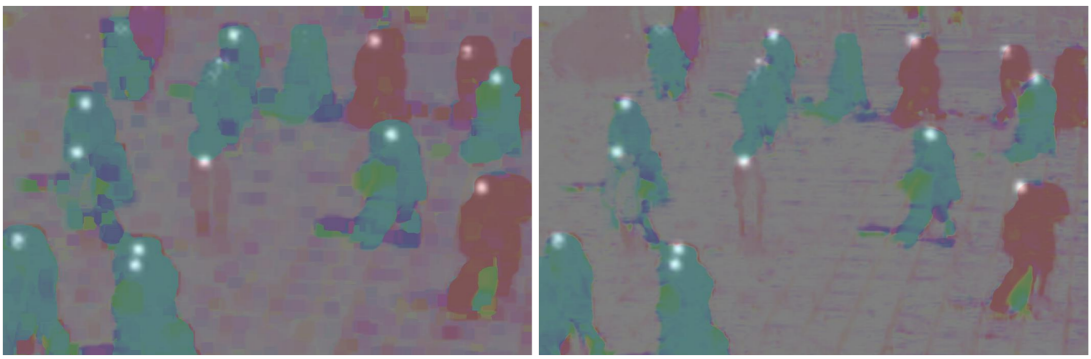


Figure 3.5: On the right a non-maxed flow estimation and on the right the flow estimation with maxing filter applied

V2, better labelling the dots in the figure, separating the figure description

Chapter 4

Implementation

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

4.1 Models

In the results four different models are compared: Two baselines and two proposed models.

4.1.1 Baseline 1

The first model is proposed in [Zhang et al., 2016]. Because this model assumes a supervised Flow Estimation, parts of the proposed method can't be used. Only the proposed model therefore is used as baseline, where the exact network is shown in figure **??**.

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.

4.1.2 Baseline 2

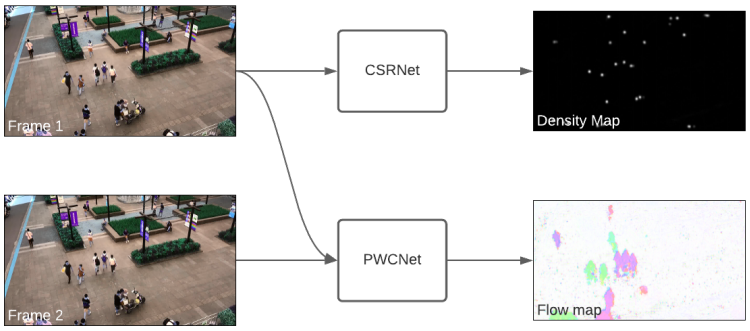


Figure 4.1: CSRNet+PWCNet baseline

This stronger baseline is a combination of two independent models (Figure 4.1). 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.

4.1.3 Unified model

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.

In V2, a pyramid decoder is used, which intermediately sends lower resolution density maps to optimize weight optimization and reduce overfitting

Show decoder per layer??

4.1.4 Flow context density map

The second model proposed enhances the feature maps with the output of the flow map. The flow enhanced model uses the first proposed model (Figure A.1) 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.

In V2 the structure is summarised in a visual graph as well, now only in appendix

Explain more in depth

4.2 LOI Counting

Explain here in more depth adding smoothing by using a region

4.3 Environment

4.3.1 Maxing filter

During experiments the maxing filter is optimized per dataset. For the Fudan-ShanghaiTech dataset a maxing filter is applied with a distance of 12px, optimized in implementation. For the UCSD dataset a maxing filter of 4px is applied.

4.3.2 Line Crossing

In the equation 3.6 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.

4.3.3 Optimizer

For all the experiments the Adam optimizer [Kingma and Ba, 2015] is used with a learning-rate of $2 \cdot 10^{-5}$. A regularization of 10^{-4} is applied.

During training, the unified loss (Equation 3.1) is used for training using a λ of 5. Which at the start focusses mainly on the density map loss and after some initial training produces an equal loss distribution between the velocity map loss and density map loss.

4.3.4 Augmentation

To augment the dataset several augmentations will be applied on the trainingssamples which are used in [Li et al., 2018] as well. 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.

TODO, when applying no cropping, less overfitting occurs at the corners, but much slower

4.3.5 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.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|C_i - P_c^{(i)}| \tag{4.1}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(C_i - P_c^{(i)})^2 \tag{4.2}$$

Rewrite, because UCSD uses a different method, which is kind of similar, but slightly different

For the ROI the MAE and the MSE are defined as equation 4.1 and 4.2. Where $P_c^{(i)}$ is the predicted density map for the given frame.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|G_l^{(i)} - P_l^{(i)}| + |G_r^{(i)} - P_r^{(i)}| \tag{4.3}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(G_l^{(i)} - P_l^{(i)})^2 + (G_r^{(i)} - P_r^{(i)})^2 \tag{4.4}$$

$$RMAE = \frac{1}{n}\sum_{i=1}^{n}\frac{|G_l^{(i)} - P_l^{(i)}| + |G_r^{(i)} - P_r^{(i)}|}{G_l^{(i)} + G_r^{(i)}} \tag{4.5}$$

For the LOI the MAE and MSE are defined as quation 4.3 and 4.4. The RMAE is simply defined as in equation 4.5. 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 5

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 labelling a tool is written and explained. Lastly all the datasets are explained in more depth.

5.1 Requirements

For training and evaluation we need two different methods of labelling. 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 labelling.

5.2 Labelling

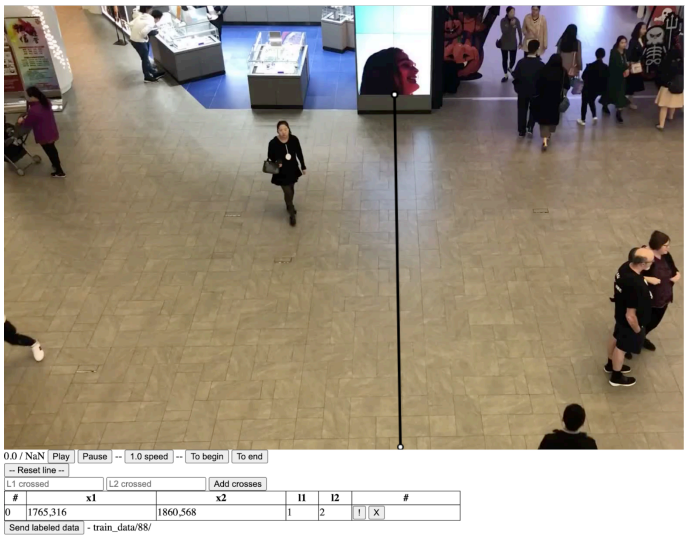
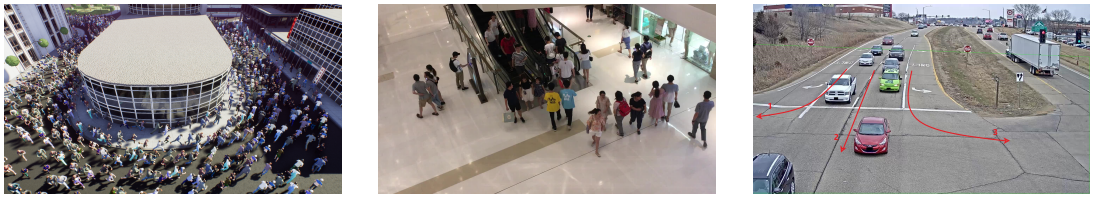


Figure 5.1: User interface of the labeller

Several Crowd Counting datasets provide sequences of frames with corresponding pedestrian labelling. However most of those datasets don't provide line crossing

labelling. Therefore I build a tool to label videos for line crossing (Figure 5.1). The labeller 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.

5.3 Datasets



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

Figure 5.2: Samples of each dataset

5.3.1 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 5.2b). 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 labelling requires the use of the custom build labeller (Figure 5.1). This is done on the 40 videos of the test set.

5.3.2 AI City Challenge

The AI City Challenge dataset is a huge dataset of cars crossing on a road. The total dataset contains 20 different scenes, but only the four busiest scenes are used for training and testing. Each scene contains 3.5 minutes of footage in 10FPS, where the first half of the scene is used as training and the second half for testing. Each vehicle is labeled as a bounding box, which can be used for the centre-point. Additionally a trajectory of each vehicle is provided, which will be used to create the Line of Interest labelling. So no extra labelling for this dataset is required. ww

5.3.3 CrowdFlow

CrowdFlow is a synthetic dataset generated using the game engine Unreal. The dataset contains 5 scenes with each a sequence with a panning camera and a fixed camera. For the experiments only 5 sequences with the fixed camera are used. Each sequences has a duration of 12.5 seconds at a FPS of 25. The CrowdFlow dataset sequences contain a large amount of pedestrians ranging between 200-1000. For each pedestrian the position and trajectory is labeled.

For each sequence 3 lines are drawn. By utilizing the present trajectory, for each line the crossed pedestrians can be counted, these crosses range between 20 and 150 per line.

Due to the sparsity of the data only 20% of the data (10% at the start and 10% at the end of the sequence) is used for training and the rest is used as testing.

During testing it appeared that the last sequence (IM05) contains errors in the labeling. Therefore the last sequences is discarded, so in total 4 sequences are used for this dataset.

Chapter 6

Results

6.1 Fudan-ShanghaiTech

Method (LOI)	MAE	MSE	RMAE
Baseline 1	-	-	-
Baseline 2	-	-	-
Proposed	3.152	10.715	0.555
Baseline 1+m	1.404	3.596	0.252
Baseline 2+m	1.422	3.474	0.259
Proposed+m	1.419	3.783	0.259
Proposed+p+m	1.397	4.166	0.255
Proposed+p+f+m	1.299	3.195	0.234
Proposed+p+f+w+m	-	-	-

Table 6.1: The LOI results for Fudan

Method (ROI)	MAE	MSE
Baseline 1	5.797	50.257
Baseline 2	2.991	15.415
Proposed	2.654	13.620
Proposed+p	2.454	11.525
Proposed+p+f	2.377	10.510
Proposed+p+f+w	-	-

Table 6.2: The ROI results for Fudan

Add the shortcuts: m=maxingfilter, f=flowfeatures, w=flowwarping, c=newCrossingLOI

For te Fudan-ShanghaiTech dataset both the LOI performance and the ROI performance is displayed in table 6.1 and table 6.2.

Looking at table 6.1. 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 velocity map is trained in a supervised manner. 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 6.2, 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.

6.2 CrowdFlow

Method (LOI)	MAE	MSE	RMAE
Baseline 1	-	-	-
Baseline 1+m	-	-	-
Baseline 2	-	-	-
Baseline 2+m	-	-	-
Proposed+m	-	-	-
Proposed+m+c	-	-	-
Proposed+w+m	-	-	-
Proposed+f+m	-	-	-

Table 6.3: The LOI results for AI City Challenge dataset

Method (ROI)	MAE	MSE
Baseline 1	-	-
Baseline 2	-	-
Proposed	-	-
Proposed+w	-	-
Proposed+f	-	-

Table 6.4: The ROI results for AI City Challenge dataset

To be done about table 6.4 and table 6.3

To be done!!!

6.3 AI City Challenge

Method (LOI)	MAE	MSE	RMAE
Baseline 1+m	14.09	273.71	0.59
Baseline 2+m	-	-	-
Proposed+m	3.89	17.97	0.24
Proposed+p+m	4.16	13.73	0.23
Proposed+p+f+m	3.76	11.45	0.21
Proposed+p+f+w+m	-	-	-

Table 6.5: The LOI results for AI City Challenge dataset

Retrain LOI Proposed+f+m+c

- It appears that the proposed LOI method isn't working perfectly. A reason for this is the sensitivity of the method on the Flow. When the prediction is off with the crossing LOI it can happen that it counts certain pixels double, because it thinks in the first pair that it crossed the line, where in reality it didn't, then it counts the pixel double. When using moving LOI it averages the speed difference. (However still weird when counted double in de Area, so wrong analysis???)

- It appears that the proposed models have the tendency to overfit much more than CSRNet. This could be because of the encoder, which is much more robust for the CSRNet than the proposed model (Where the encoder is not designed for general usage).

To be done about table 6.5

To be done!!!

6.4 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.

FPS	MAE	RMAE
25	-	-
12.5	-	-
5	-	-
2.5	-	-
1	-	-

Table 6.6: Optimal FPS

Method	FPS	ms	RMAE
Baseline 2	-	-	-
Baseline 2+m	-	-	-
Proposed+m	-	-	-
Proposed+f+m	-	-	-
Proposed+m+c	-	-	-
Proposed+w+m+c	-	-	-
Proposed+f+m+c	-	-	-
Proposed+m+c+o	-	-	-
Proposed+f+m+c+o	-	-	-

Table 6.7: Processing time

Looking at table 6.6 it shows that a higher FPS does not always improve performance. The table shows that the optimal results are made at a frame rate of 5 FPS. Which could be caused by the instability of the Flow Estimation on a very high frame rate. At a low frame rate people could walk in a non-linear way or it could enhance errors in the density map.

Table 6.7 shows that the proposed model with maxing is almost double the speed in performance. Additionally when an optimized version is used where further optimizations for unified models are applied the model performs even better without degradation of performance. Also the flow enhancing shows only a slight increase in processing time.

6.5 Flow estimation impact

Qualitative comparison (Better show the realigning problem and how that this is solved by the maxing filter)

Show some overfitting problems on the corners, but that this isnt very important, because most of the LOI's are in the centre of the image and not at the corners. (Could be fixed with better cropping in augmentation)

Chapter 7

Conclusion

In this thesis a new method for Crowd Crossing Counting is presented which can be used multi-domain as well.

All results on the datasets show a clear benefit for a newly created unified model which is focussed on unsupervised flow estimation in comparison to [Zhao et al., 2016].

The first proposed model is performing equally well as the strong baseline. Which was predicted, due to use of the same principles the model design was based on.

The model with flow context clearly performs better on all models, which suggests that flow context indeed pushes the model indirectly to focus more on the moving pedestrians.

The Fudan-Shanghai and UCSD datasets show that the usage of realigning the density map and velocity map is crucial to perform well. The maxing filter is a good solution to artificially align the density map and the velocity map.

The AI City Challenge dataset show that the proposed method is multi-domain as well and performing seemingly well. Additionally the dataset shows that the use of realigning is of less usage for line crossing with objects where the labelling is done in the middle of the object and of sufficient size.

The results are promising for Crowd Counting for used in real world applications. Multiple streams can be processed in real time running on a single GPU. However the method is currently run on a large GPU, which is often still difficult to store. However with the increase in efficiency and performance of current GPU's, the model could soon be running on much smaller equipment.

All in all, even though the proposed model doesn't show state-of-the-art performance on the UCSD benchmark, the proposed model is an interesting new approach to Crowd Crossing Counting.

7.1 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.

— Focussing on even more efficient usage of the method (Aligning is expensive now)

— No real comparison or research is done on other realigning methods. Currently some parameters need to be set, a trainable method would be interesting for further research.

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.

[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.

[Kingma and Ba, 2015] Kingma, D. P. and Ba, J. L. (2015). Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pages 1–15.

[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.

[Shi et al., 2019] Shi, Z., Mettes, P., and Snoek, C. G. (2019). Counting with focus for free. In *ICCV*, pages 4200–4209.

[Shi et al., 2018] Shi, Z., Zhang, L., Liu, Y., Cao, X., Ye, Y., Cheng, M.-M., and Zheng, G. (2018). Crowd counting with deep negative correlation learning. In *CVPR*.

[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-Decem: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

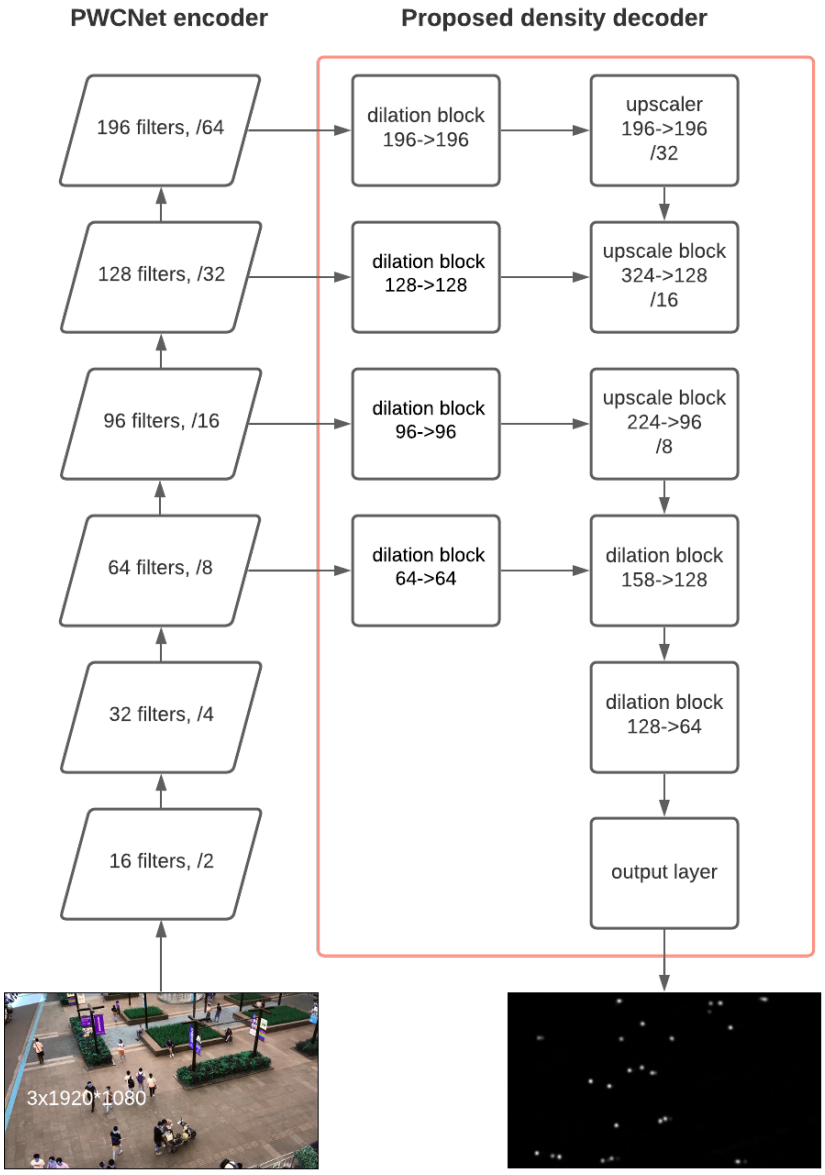


Figure A.1: Full decoder for density prediction