Crowd counting with machine learning techniques

Seminar

Image Based Biometrics 2020/21, Faculty of Computer and Information Science, University of Ljubljana

Maša Kljun and Matija Teršek
mk2700@student.uni-lj.si, mt2421@student.uni-lj.si

***Abstract*—Crowd counting has a range of applications and it is an important task as it can help with the prevention of accidents such as crowd crushes and stampedes in political protests, concerts, sports, and other social events. A lot of algorithms have been proposed for tackling problems of crowd counting and in this paper we focus on five convolutional neural network (CNN) approaches - CSRNet, Bayesian Crowd Counting, DM-Count, SFA-Net, and SGA-Net. We briefly desribe the models and train and evaluate them ourselves with the help of three well known crowd image datasets, ShanghaiTech part A, part B, and UCF-QNRF. Furthermore, we propose the upgrade of one model and compare its results to the original one, showing that it is in fact an improvement. [1] In general we see that the results obtained on the ShanghaiTech dataset are better as opposed to those obtained on UCF-QNRF due to smaller and cropped images. We show that SFA-Net and DM-Count perform the best on ShanghaiTech part A dataset, with the first and SGA-Net having the best performance on ShanghaiTech part B, and the latter having the best performance on the QNRF dataset.**

I. Introduction

Automatic estimation of a number of people in a crowd is an important technique with applications in many fields. Political protests, rallies, concerts, religious events, etc., are just some of the situations that can benefit from the automatic crowd counting, since having a good estimate of the crowd can help prevent crowd crushes, stampedes, and other accidents. Furthermore, in the light of the recent pandemic of the COVID-19, crowd counting and crowd analysis can help prevent the spread of the virus by ensuring enough physical distance between people in some usually crowded public places, such as stores, cinemas, recreational areas, etc.

In addition to the mentioned applications, crowd counting is popular as it can be easily extended to counting tasks in other fields. Some of them include counting vehicles for traffic control [11], monitoring discarded fish catch and counting penguins for environmental control [2, 3], counting leafs for plant phenotyping [1], and estimating the number of cells in microscopic images [7]. Crowd counting is crucial in such tasks as it automates and speeds up otherwise tedious processes.

Because of the wide variety of applications of crowd counting methods, a lot of research has been made and many different algorithms have been proposed. Different approaches to crowd counting exist, and they can be roughly divided into 3 groups - detection, regression, and density based. While some related works include overviews of existing crowd analysis methods [4, 9, 12, 13, 17], the other focus more on discovering the new approaches [8, 10, 14, 16, 19].

In this paper we focus on CNN based approaches, as they recently began to gain in the popularity. We briefly describe and provide key features of five state-of-the-art models. Unlike some of the related works, who only gather the results from author's papers, we try to train and evaluate the models ourselves on three popular crowd counting datasets. Furthermore, we propose an improvement for one of the models and compare it to the others.

This paper is organized as follows: In Chapter II we provide the most common approaches to crowd counting, in Chapter III we describe five state-of-the-art CNN models and our suggested improvement, and in Chapter IV we describe the three datasets on which we evaluate the models and discuss the results of our evaluation.

II. Crowd counting approaches

The goal of crowd counting methods is to determine the number of people present in a particular area. There exist many different approaches of doing this and we can divide them into 3 main categories:

1. Detection based approaches:
This is the most straight-forward approach that can use whole bodies (Monolithic detection) or just parts of it (Part-based detection), e.g., the combination of head and hands. Methods in this group use features such as Haar wavelets or histogram of oriented gradient (HOG) to represent the body, and then use a classifier with the sliding window approach across the image to detect person candidates. The used classifiers are usually linear (e.g., linear support vector machine (LSVM)), as the non-linear (e.g., support vector machine (SVM)) lack the detection speed. The drawback of detection based approaches is that they fail in high occlusion situations or in highly crowded spaces [9].

2. Regression based approaches:
The idea of this group's methods is not to count individuals, but to estimate the crowd density, which is specifically useful in more crowded places. Methods in this group first extract low-level features (e.g., foreground or edges). Then, with the help of a linear regression model, a mapping between low-level features and people count is made. The drawback of regression based approaches is that when the same object is placed in different depths in the image, the values of features extracted from those objects can vary upon the depth of where the object was placed. However, this problem can be tackled by geometric correction [9].

3. Density based approaches:
The idea of this group's methods in its most simplistic form is to obtain a density map from an image and then integrate it in order to get the estimation of people in the image. However, the methods differ in the choice of a training loss function (e.g., squared error between the predicted density values and the ground truth) and in the choice of a density map prediction method (e.g., with the help of a linear model) [6].

In 2015 the pioneering work with deep networks in crowd

[1] Source code and pretrained weights are available at: https://github.com/tersekmatija/crowd-counting-cnns.

counting was introduced ([15]), introducing CNN approaches to the crowd counting. Since then many of CNN based approaches were proposed. The basic idea behind CNN based approaches is that they normally try to predict the density map from the input image and infer the count from it. Having said that, they could be classified under density based approaches. Models that are based on CNNs differ in the usage of different backbones (e.g., VGG-16, VGG-19, Inception v3 - see Figure 1), loss functions, additional maps (e.g., attention map), and model structure (e.g., single or multi column). We describe five CNN models for crowd counting along with their key features and one improved model in the next chapter.

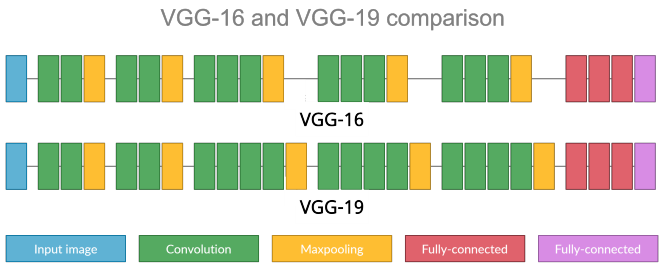

Figure 1: Figure shows difference between VGG-16 and VGG-19 architecture, which are often used as a backbone in CNN crowd counting models. They also appear in models described in this paper. Additionally, we describe a model which uses Inception v3 as a backbone, but we do not show its architecture here, as it is more complicated.

III. CNN models

In this section we shortly describe each of the models. Note that we put the main focus on their features, where they differ the most from each other.

A. CSRNet

The architecture of this model is divided into 2 parts: a CNN at the front-end and a dilated CNN at the back-end. The basis of CSRNet front-end is build on VGG-16 model with the fully-connected layers removed [8]. Ten layers of VGG-16 are kept, with only three pooling layers instead of five. The back-end consists of six dilated convolutional layers, for which the authors suggest that it represents a good alternative to the pooling layers. Dilated convolution can be used instead of the pooling layer, since it mantains the resolution of feature map and contains more detailed information. Another 1 convolutional layer is added as the output layer.

Authors suggest different models, which are determined by different back-end settings that vary in the dilation rate. We use the model B in our experiments, as it is the most successful [8], where the dilation rate is set to 2 for all the back-end layers.

1) Dilated convolution: The idea of the dilated convolution is that is uses sparse kernels, which enlarge the receptive field. Same can be achieved by adding more convolutional layers, however, that increases the computational cost.

For input $x(m,n)$ and filter $w(i,j)$, of length M, width N, and the dilation rate r ($r=1$ results in a normal convolution), we can define output $y(m,n)$ of the dilated convolution as

$$\sum_{i=1}^{M}\sum_{j=1}^{N} x(m+r\times i,\ n+r\times j)w(i,j) \tag{1}$$

2) Loss function and training: Loss function is derived from the Euclidean distance between the ground truth and estimated density map. The loss function is defined as

$$\mathcal{L}=\frac{1}{2N}\sum_{i}^{N}||D_i^{est}-D_i^{gt}||_2^2, \tag{2}$$

where N is the size of the training batch, D_i^{est} the density map generated by the CSRNet, and D_i^{gt} the ground truth density map of the input image.

In training the first 10 convolutional layers are fine-tuned from a trained VGG-16. Initial settings for other layers are set with the help of a Gaussian distribution with 0.01 standard deviation, and stochastic gradient descent (SGD) with rate $1e-6$ is applied during training.

B. Bayesian crowd counting

The Bayesian model uses VGG-19 as the backbone, with the last pooling and the subsequent fully connected layers removed. The output of the backbone is upsampled to $\frac{1}{8}$ of the input image size by bilinear interpolation and fed to a regression header. The regression header consists of two 3×3 convolutional layers, one with 256 and the other with 128 channels, and one 1×1 convolutional layer. The produced output is a density map [10].

Bayesian crowd counting model differs from other CNN based models in the utilization of a loss function. Opposed to the previous models, which use a Gaussian kernel to obtain the ground truth density map and define loss function as a sum of pointwise distances between ground truth and estimated density maps, it uses a novel Bayesian loss function.

1) Bayesian Loss function and training: We can derive the loss function as follows. Let x be a random variable describing the spatial location, and y be a random variable representing the annotated head point. Let $m=1,...,M$ where M is the number of pixels in the density map and let $n=1,...,N$, where N is the total crowd count. Let z_n be a head position and y_n be a corresponding label. The likelihood function of location x_m given the label y_n can be defined as

$$p(x_m|y_n)=N(x_m;z_n,\sigma^2 1_{2\times2}), \tag{3}$$

where $N(x_m;z_n,\sigma^2 1_{2\times2})$ is a 2D Gaussian distribution evaluated at x_m, with the mean at the annotated point z_n and an isotropic covariance matrix $\sigma^2 1_{2\times2}$.

Using Bayes we can then compute

$$p(y_n|x_m)=\frac{p(x_m|y_n)p(y_n)}{p(x_m)}=\frac{N(x_m;z_n,\sigma^2 1_{2\times2})}{\sum_{n=1}^{N}N(x_m;z_n,\sigma^2 1_{2\times2})}. \tag{4}$$

The Bayesian loss function can be defined as

$$\mathcal{L}^{\text{Bayes}}=\sum_{n=1}^{N}\mathcal{F}(1-E[c_n]), \tag{5}$$

where $\mathcal{F}$ is a distance function (ℓ_1) and $E[c_n]$ is the expected value of a total count associated with y_n, that can be computed as

$$E[c_n]=\sum_{m=1}^{M}p(y_n|x_m)D^{est}(x_m). \tag{6}$$

When inferring, the total count is just a sum over an estimated density map.

Additionally, authors introduce the background pixel modeling for background pixels that are far away from any of the

annotation points. They introduce an additional background label $y_0 = 0$ in addition to the head labels, as it makes no sense to assign the background pixels to any of the head labels. The posterior label probability is then rewritten and additional expected count for the entire background $E[c_0]$ is introduced. The deeper analysis of the background pixel modeling is beyond the scope of this paper, as we want to provide relatively short and concise descriptions of the models, but it is worth noticing that it defines a new enhanced loss function

$$\mathcal{L}^{Bayes+} = \sum_{n=1}^{N} \mathcal{F}(1 - E[c_n]) + \mathcal{F}(0 - E[c_0]). \tag{7}$$

MSRA initializer is used for the initialization of the regression header, whereas the backbone is pre-trained on ImageNet. Parameters are updated with the help of the Adam optimizer with an initial learning rate $1e-5$.

C. Bayesian CSRNet

We implement a new model based on the CSRNet and Bayesian crowd counting loss function and pixel modeling, with the goal of improving the basic CSRNet. The basic structure of our model is the same as the one of the CSRNet, described in Subsection III-A. We use the first ten layers of the VGG-16 with 3 pooling layers for the front-end, 6 convolutional layers with the dilation rate set to 2 as the back-end, and an additional 1×1 layer as the output layer.

1) Loss function and training: Instead of CSRNet's loss function provided in Equation 2, we use the Bayesian+ loss function described in Equation 7.

The weights are initialized in the same way as in the CSRNet. The first 10 convolutional layers are fine-tuned from a trained VGG-16, whereas initial settings for other layers are obtained with the help of a Gaussian distribution with 0.01 standard deviation. Parameters are updated with the help of the Adam optimizer with an initial learning rate of $1e-6$.

D. SFANet

The next model we analyse is SFANet [19]. It uses the first 13 layers of a pre-trained VGG-16-bn (VGG-16 with batch normalization) as the front-end feature map extractor. It is suitable as it has a strong ability to represent features and can be easily concatenated by the back-end dual path networks. Four source layers (conv2-2, conv3-3, conv4-3, and conv5-3) are then connected to a dual multi-scale fusion networks with attention (density map path and attention map path), which represent the back-end. Attention map path is incorporated to tackle the background noise and non-uniformity of crowd distributions.

1) Loss function and training: In most models an Euclidean loss is used for measuring estimation error, which is defined as:

$$\mathcal{L}^{\text{DEN}} = \frac{1}{N} \sum_{i=1}^{N} \|D_i^{est} - D_i^{gt}\|^2, \tag{8}$$

where D_i^{est} is the estimated density map of i-th input image, D_i^{gt} represents the ground truth density map, and N is the batch size. SFANet also uses the described loss function. In addition, the model uses the attention map loss function, a binary class entropy defined as

$$\mathcal{L}^{\text{ATT}} = -\frac{1}{N} \sum_{i=1}^{N} (A_i^{gt} \log(P_i) + (1 - A_i^{gt}) \log(1 - P_i)), \tag{9}$$

where A_i^{gt} is the attention map ground truth, and P_i probability of each pixel in predicted attention map activated by sigmoid function.

The unified loss function is then defined as

$$\mathcal{L} = \mathcal{L}^{\text{DEN}} + \alpha \mathcal{L}^{\text{ATT}}, \tag{10}$$

with α weighting weight set to 0.1.

The first 13 layers of a pre-trained VGG-16-bn are applied as the front-end feature extractor. Other parameters are randomly initialized with a Gaussian distribution with a standard deviation 0.01. Parameters are updated with the help of Adam optimizer with learning rate of $1e-4$ and weight decay of $5e-3$.

2) Ground truths: Density map groundtruth D^{gt} is obtained similarly as in most models, with the use of Gaussian kernels.

Attention map groundtruth is obtained from D^{gt} and Gaussian kernel as

$$\begin{aligned} \mathbb{Z} &= D_i^{gt} \times G_{\mu,\sigma^2}(x), \\ A_i^{gt}(x) &= \begin{cases} 0, & x < thresh \\ 1, & x \geq thresh \end{cases}, \forall x \in \mathbb{Z}, \end{aligned} \tag{11}$$

with *thresh* set to 0.001.

E. DM-Count

DM-Count model considers crowd counting as a distribution matching problem [14]. The architecture of the model is based on the VGG-19 and is the same as in the Bayesian Crowd Counting model (see Subsection III-B). Different to the previous models, who use density map estimations that are computed with the help of Gaussian kernels, DM-Count can preprocess ground truth annotations without the use of a Gaussian. Instead it uses Optimal Transport (OT) to measure the similarity between the normalized predicted density map and the normalized ground truth density map. OT computation is then stabilized with the help of a Total Variation (TV) loss.

1) Loss function and training: The loss function is the combination of the counting loss, optimal transport loss, and the total variation loss. Let $z \in \mathbb{R}_+^n$ be a vectorized binary map for dot annotation, and $\hat{z} \in \mathbb{R}_+^n$ a vectorized predicted density map.

2) Counting loss:

$$\mathcal{L}^{\text{COUNT}}(z, \hat{z}) = |\|z\|_1 - \|\hat{z}\|_1|, \tag{12}$$

where $\|\cdot\|$ denotes the L_1 norm.

3) Optimal transport loss: Since z and $\hat{z}$ are both unnormalized density functions, they can be turned into the probability density functions (PDF) with dividing them by their respective total mass. Optimal transport loss is then defined as

$$\begin{aligned} \mathcal{L}^{\text{OT}}(z, \hat{z}) &= \mathcal{W}(\frac{z}{\|z\|_1}, \frac{\hat{z}}{\|\hat{z}\|_1}) \\ &= \left\langle \alpha^*, \frac{z}{\|z\|_1} \right\rangle + \left\langle \beta^*, \frac{\hat{z}}{\|\hat{z}\|_1} \right\rangle, \end{aligned} \tag{13}$$

where $\mathcal{W}$ is a Monge-Kantorovich's Optimal Transport cost (see [14] for the definition), with α^* and β^* being solutions of the optimal transport problem.

Authors suggest the use of OT instead of some other measure of similarity between two PDFs, such as Kullback-Leibler divergence or Jensen-Shannon divergence, as it provides a valid gradient to train a network. The gradient with respect to $\hat{z}$ can be obtained as

$$\frac{\partial \mathcal{L}^{OT}(z, \hat{z})}{\partial \hat{z}} = \frac{\beta^*}{||\hat{z}||_1} - \frac{\langle \beta^*, \hat{z} \rangle}{||\hat{z}||_1^2}, \tag{14}$$

which can be back-propagated to learn the parameters of the density estimation network.

4) Total variation loss: OT loss is optimized with Sinkhorm algorithm for approximating α^* and β^* in each training iteration. Due to this optimization, OT loss approximates well more dense areas, but it performs poorer for the low density areas. To cope with that, Total variation loss is additionally used and can be defined as

$$\mathcal{L}^{\text{TV}}(z, \hat{z}) = \frac{1}{2} \left\| \frac{z}{\|z\|_1} - \frac{\hat{z}}{\|\hat{z}\|_1} \right\|_1. \tag{15}$$

F. SGANet

The SGANet model is the first model that investigates Inception-v3 as a backbone network instead of VGG-16, VGG-19, or ResNet, as in the most state-of-the-art models [16]. Fully-connected layers and two maxpooling layers are removed. Before the last Inception Module an upsampling layer is added, which is connected to both, the attention layer and the last Inception Module. Attention layer's output is then applied to the feature maps generated by the last Inception Module.

1) Loss function and training: The authors introduce a novel curriculum loss strategy to address the issues caused by extremely dense regions. This is a strategy of model learning where easy examples are selected at the beginning of the training and more difficult ones are added to the training set gradually. A threshold is used for determining the difficulty score, where density map pixels with higher values than the threshold have higher difficulty scores, since such pixels are within the regions of denser crowds. The whole training set is used throughout the training process, however, the threshold is first set to a low value and then gradually increased, which turns difficult pixels into easy ones so that they contribute more to the training.

The loss function is defined as a sum of two loss functions:

$$\mathcal{L} = \mathcal{L}^{\text{DEN}} + \lambda \mathcal{L}^{\text{SEG}}, \tag{16}$$

where λ is a hyper-parameter set to 20. The density map loss can be calculated as

$$\mathcal{L}^{\text{DEN}} = \frac{1}{2N} \sum_{i=1}^{N} \|\hat{M}_i^{den} - M_i^{den}\|_F^2 \tag{17}$$

and segmentation map loss is defined as the cross-entropy loss as

$$\begin{aligned} \mathcal{L}^{\text{SEG}} = -\frac{1}{N} \sum_{i=1}^{N} \| M_i^{seg} \odot \log(\hat{M}_i^{seg}) \\ + (1 - M_i^{seg}) \odot \log(1 - \hat{M}_i^{seg}) \|_1, \end{aligned} \tag{18}$$

where $\|\cdot\|_1$ denotes the elementwise matrix norm, $\odot$ denotes elementwise multiplication of two same-size matrices, and M^{seg} and M^{den} represent ground truth (without hat) and estimated (with hat) segmentation and density maps.

2) Ground truth: Ground truth density map M^{den} is obtained using a Gaussian kernel with fixed σ. Segmentation map ground truth is obtained similarly, but as

$$M^{seg}(x) = \sum_{i=1}^{N} \delta(x - x_i) * J_n(x), \tag{19}$$

where $J_n(x)$ is an all-one matrix of size $n \times n$ centered at the position x. Wang and Breckon [16] set $n = 25$.

Model uses the Adam optimizer for updating the parameters, where the initial learning rate is set to $1e-4$ and reduced by a factor of 0.5 after every 50 epochs. The weights of the Inception layers are loaded from a pre-trained Inception-v3 model.

IV. Experiments and Results

A. Data

We test the described models on the following three publicly available datasets.

1) ShanghaiTech Dataset: ShanghaiTech consists of two datasets/parts - part A and part B [18]. Part A contains 482 images randomly downloaded from the internet, containing highly congested scenes. It contains a total of 241,667 annotated people, with a 501 average per image, and 3139 maximum. It comes split into a train and a test set, containing 300 and 182 images, respectively. Images in this dataset are challenging to count, as they contain extremely congested scenes, varied perspective, and unfixed resolution. Figure 2 shows some examples of ShanghaiTech part A train set images.

Figure 2: Figure shows 4 randomly chosen images from the ShanghaiTech part A train set. We can see that the images are cropped to contain dense crowds only.

Part B contains 716 images that are taken from the busy streets of metropolitan areas of Shanghai. Images are of fixed size and contain total of 88,488 annotated people, with a 124 average per image, and 578 maximum. Same as the part A, it is already split into a train and a test set, containing 400 and 316 images, respectively. As images are captured in metropolitan areas, they contain relatively sparse crowds and include streets, buildings, vegetation, and sometimes rivers as well. In Figure 3 we show some examples of the ShanghaiTech part B train set images.

2) UCF-QNRF Dataset: UCF-QNRF is among the newest and the largest datasets for crowd counting problems [5]. It consists of 1525 images and contains a total of 1,251,642 annotated people, with a 815 average, and 12,865 maximum. It is split into a train and a test set, containing 1201 and 334 images, respectively. Dataset contains images with congested scenes with a diverse set of viewpoints, densities, and lighting variations. Different from the ShanghaiTech part A, which contains images with dense crowds that are cropped to contain crowds only,

Figure 3: Figure shows 4 randomly chosen images from the ShanghaiTech part B train set. We can see that the images contain relatively sparse crowds. The background often consists of buildings and vegetation, but can also include rivers as seen in the top left image.

images from this set also contain buildings, vegetation, sky, and roads, as they are present in realistic scenarios captured in the wild, making the dataset more realistic but also more difficult to count. Figure 4 shows some examples of UCF-QNRF train set images.

Figure 4: Figure shows 4 randomly chosen images from the UCF-QNRF train set. We can see that the images are more realistic than the images from the ShanghaiTech part A, as they include not only crowds but also buildings, sky, and vegetation.

B. Evaluation metrics

We use Mean Absolute Error (MAE) and Mean Squared Error (MSE) for the evaluation and they are defined as follows

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|C_i - C_i^{GT}|, \tag{20}$$

$$\text{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}|C_i - C_i^{GT}|^2}, \tag{21}$$

where n is a number of images, C_i represents the inferred count, and C_i^{GT} represents the ground truth count.

C. Evaluation

We train and test the models on the three mentioned datasets - ShanghaiTech part A, part B, and UCF-QNRF. In addition to training and evaluating models separately for the three datasets, we also include the results of training the model with ShanghaiTech part A train set and evaluating it with UCF-QNRF test set. We show the obtained MAE and MSE in Table I.

We can see that the best results are in general obtained on the ShanghaiTech part B (SHB) dataset, which is expected due to the low average count per image and relatively sparse crowds. We see that the best results are obtained by the SFA-Net (7.05 MAE) and SGA-Net (11.48 MSE), followed closely by the DM-Count (7.68 MAE). The worst performance is given by the CSRNet (11.27 MAE), which is outperformed by our improved model Bayesian CSRNet (8.48 MAE) and Bayesian Crowd Counting model (8.27 MAE).

We see that the results on ShanghaiTech part A (SHA) are better than those on the UCF-QNRF due to the smaller dataset and smaller and less complicated images. We see that the best results on SHA dataset are obtained by SFA-Net and DM-Count. While the first has a lower MAE (59.58), the second has lower a MSE (98.56). They are closely followed by SGA-Net (61.58 MAE). The worst performance is obtained by CSRNet (75.44 MAE), however, we show that our Bayesian CSRNet model is in fact an improvement of the original CSRNet, with MAE of 66.92.

Due to the bigger size of the images from the QNRF dataset, we are only able to train and evaluate three models, as some have far too slow preprocessing, while some take too much disk space. However, we see that DM-Count once again performs the best (88.97 MAE), and is closely followed by Bayesian Crowd Counting (90.43 MAE). Out of the three, our improved Bayesian CSRNet performs the worst (103.94 MAE).

Due the problems with the QNRF dataset, we, in addition to the evaluation of the models on SHA, SHB, and QNRF, also show the results of training the model on SHA train set and evaluating it with QNRF test set, since they both contain relatively dense crowds. We see that the overall results here are much worse due to the models being trained on images that are cropped to contain crowds only, not including buildings and vegetation in the background. As images in the test set include those objects in the backgrounds, models could misinterpret them and count them as a crowd. The best results are given by the Bayesian Crowd Counting model (138.39 MAE), followed relatively closely by DM-Count (141.43 MAE) and our Bayesian CSRNet (145.03 MAE). The worst performance is once again achieved by the CSRNet (199.54 MAE).

Note that some results differ from the results reported in the author's papers. Some results are slightly influenced by the random seed, as fixed random seed would make the training time too long. Some authors also use other implementations in their papers (such as CSRNet, whose authors provide two official implementations - one in Pytorch and one in Caffe).

We were unable to train some models due to the computational limitations on the QNRF dataset. Some models preprocess or crop the data differently and significantly bigger images from the QNRF occupy too much disk space or cause out of memory error on the GPU. However, the training is too slow for the CPU. Preprocessing times might also be long. As authors do not provide the information on how to avoid these issues and we do not want to change the models in any way

that would lead to worse results, we denote such cases as "/" in Table I.

We show the results of our improved model in Figures 5 and 6. In the first figure we show the input images from the ShanghaiTech part A and part B test set, and predicted density maps and inferred counts on a model trained on ShanghaiTech datasets. In the second figure we show the input image from the UCF-QNRF test set and predicted density maps and inferred counts on models trained on UCF-QNRF and ShanghaiTech datasets.

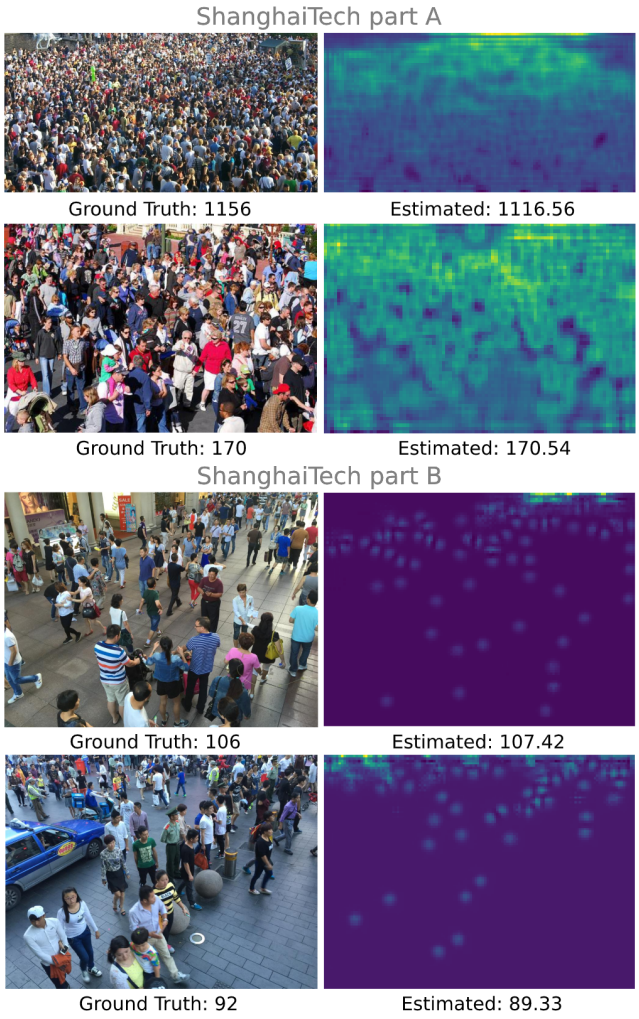

Figure 5: Top left row figures show input images from ShanghaiTech part A test dataset with 1156 and 170 annotated people. The top right row figures show the predicted density maps obtained by our Bayesian CSRNet, trained on ShanghaiTech part A. The estimated counts are 1116.56 and 170.54. Similar holds for the bottom two rows. The left column contains images taken from ShanghaiTech part B test dataset, containing 106 and 92 annotated people. The bottom right row shows the density map outputs of our Bayesian CSRNet with estimates of 107.42 and 89.33 people.

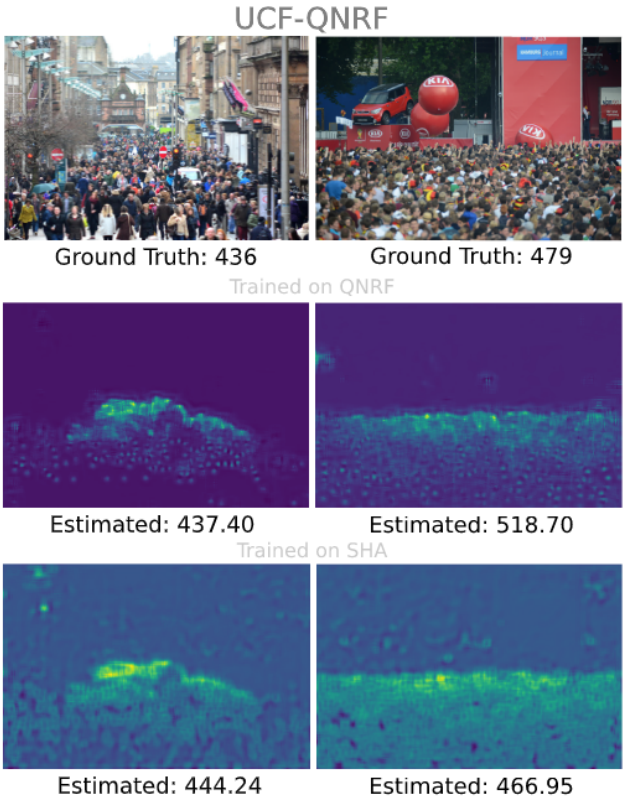

Figure 6: The upper two images show input images from UCF-QNRF with 436 and 479 annotated people. The bottom 4 figures show the predicted density maps obtained by our Bayesian CSRNet trained on UCF-QNRF (middle row) and on ShanghaiTech part A (bottom row). We see that the middle density maps are clearer, as the model here is trained on similar images that also contain building and streets, and it can better differ between them and the crowds. We also see that the inferred result is slightly better on the model trained on UCF-QNRF for the left column, but the one trained on SHA is slightly better in the right column.

V. Conclusion

In this paper we were focused on crowd counting techniques that use CNNs. We reviewed definitions and provided concise descriptions of 5 CNN based models - CSRNet, Bayesian Crowd Counting, DM-Count, SFA-Net and SGA-Net. In addition we trained and evaluated the models ourselves, contrary to many other related works who just provided evalution results from author's papers. We evaluated the models on ShanghaiTech part A dataset, ShanghaiTech part B dataset, and UCF-QNRF dataset. Additionally, we wanted to see how good the results are when training the model on one dataset (ShanghaiTech part A) and evaluating it on another (UCF-QNRF). We saw that the best overall results are those obtained on ShanghaiTech part B dataset, as models work better on images that are less complicated or have less dense crowds. The best results in terms of MAE on the ShanghaiTech part A were obtained with the SFA-Net model, followed closely by the DM-Count model. The first also performed best on the ShanghaiTech part B, and the latter also performed best on the UCF-QNRF dataset. In terms of MSE, SGA-Net outperforms the SFA-Net on ShanghaiTech part B. The results of training the models on one dataset and evaluating them on the other were less good, however, that was expected due to the smaller train set with images that were cropped to contain crowds only, whereas the images from the test set also included buildings, sky, and vegetation.

In addition to the evaluation of the 5 mentioned models, we also suggested an improvement of the CSRNet. We tried to

Datasets	SHA		SHB		QNRF		QNRF on SHA	
Method	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
CSRNet	75.44	113.55	11.27	19.32	/	/	199.54	319.09
Bayesian CSRNet	69.46	111.73	8.48	13.55	103.94	186.22	139.83	260.59
Bayesian Crowd Counting	66.92	112.07	8.27	13.56	90.43	161.41	**138.39**	**256.81**
DM-Count	61.39	**98.56**	7.68	12.66	**88.97**	**154.11**	141.43	260.23
SFA-Net	**59.58**	99.43	**7.05**	12.18	/	/	170.29	365.59
SGA-Net	61.58	101.59	7.60	**11.48**	/	/	/	/

Table I: In this table we show the evaluation of the models in terms of MAE and MSE on different datasets. The best results are marked in bold. We see that SFA-Net and DM-Count perform the best on ShanghaiTech part A (SHA), with the first giving the best performance on ShanghaiTech part B (SHB), and the latter giving the best performance also on the UCF-QNRF (QNRF). In terms of MSE, SGA-Net outperforms the SFA-Net on the SHB dataset. Bayesian Crowd Counting yields the best results when trained on SHA and evaluated on QNRF. We also show that our combination of Bayesian Crowd Counting model and CSRNet, Bayesian CSRNet, is in fact an improvement of the original CSRNet model. "/" denotes situations where we could not execute the training due to the computational limitations.

implement a new model based on the CSRNet and a Bayesian crowd counting loss function and pixel modeling. We showed that the new model is in fact an improvement of the original model.

Due to the computational limitations we were unable to train/evaluate some models on the QNRF dataset. For the future work we suggest the investigation of possible solutions. Since many datasets exist, we also suggest the evaluation of the models on other datasets (e.g., NWPU). SGA-Net also shows a possible investigation field, as it uses Inception-v3 model instead of VGG-16 or VGG-19, and yet still shows very promising results.

References

[1] Aich, S. and Stavness, I. (2017). Leaf counting with deep convolutional and deconvolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2080–2089.

[2] Arteta, C., Lempitsky, V., and Zisserman, A. (2016). Counting in the wild. In *European conference on computer vision*, pages 483–498. Springer.

[3] French, G., Fisher, M., Mackiewicz, M., and Needle, C. (2015). Convolutional neural networks for counting fish in fisheries surveillance video.

[4] Gao, G., Gao, J., Liu, Q., Wang, Q., and Wang, Y. (2020). Cnn-based density estimation and crowd counting: A survey. *arXiv preprint arXiv:2003.12783*.

[5] Idrees, H., Tayyab, M., Athrey, K., Zhang, D., Al-Maadeed, S., Rajpoot, N., and Shah, M. (2018). Composition loss for counting, density map estimation and localization in dense crowds. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–546.

[6] Kang, D., Ma, Z., and Chan, A. B. (2018). Beyond counting: comparisons of density maps for crowd analysis tasks—counting, detection, and tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(5):1408–1422.

[7] Lempitsky, V. and Zisserman, A. (2010). Learning to count objects in images. *Advances in neural information processing systems*, 23:1324–1332.

[8] 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.

[9] Loy, C. C., Chen, K., Gong, S., and Xiang, T. (2013). Crowd counting and profiling: Methodology and evaluation. In *Modeling, simulation and visual analysis of crowds*, pages 347–382. Springer.

[10] Ma, Z., Wei, X., Hong, X., and Gong, Y. (2019). Bayesian loss for crowd count estimation with point supervision. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6142–6151.

[11] Onoro-Rubio, D. and López-Sastre, R. J. (2016). Towards perspective-free object counting with deep learning. In *European Conference on Computer Vision*, pages 615–629. Springer.

[12] Saleh, S. A. M., Suandi, S. A., and Ibrahim, H. (2015). Recent survey on crowd density estimation and counting for visual surveillance. *Engineering Applications of Artificial Intelligence*, 41:103–114.

[13] Sindagi, V. A. and Patel, V. M. (2018). A survey of recent advances in cnn-based single image crowd counting and density estimation. *Pattern Recognition Letters*, 107:3–16.

[14] Wang, B., Liu, H., Samaras, D., and Nguyen, M. H. (2020). Distribution matching for crowd counting. *Advances in Neural Information Processing Systems*, 33.

[15] Wang, C., Zhang, H., Yang, L., Liu, S., and Cao, X. (2015). Deep people counting in extremely dense crowds. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 1299–1302.

[16] Wang, Q. and Breckon, T. P. (2019). Segmentation guided attention network for crowd counting via curriculum learning. *arXiv preprint arXiv:1911.07990*.

[17] Zhan, B., Monekosso, D. N., Remagnino, P., Velastin, S. A., and Xu, L.-Q. (2008). Crowd analysis: a survey. *Machine Vision and Applications*, 19(5-6):345–357.

[18] Zhang, Y., Zhou, D., Chen, S., Gao, S., and Ma, Y. (2016). Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 589–597.

[19] Zhu, L., Zhao, Z., Lu, C., Lin, Y., Peng, Y., and Yao, T. (2019). Dual path multi-scale fusion networks with attention for crowd counting. *arXiv preprint arXiv:1902.01115*.