Crowd Counting with Decomposed Uncertainty

Min-hwan Oh
Columbia University
New York, NY 10027
m.oh@columbia.edu

Peder A. Olsen
IBM Research
Yorktown Heights, NY 10598
pederao@us.ibm.com

Karthikeyan Natesan Ramamurthy
IBM Research
Yorktown Heights, NY 10598
knatesa@us.ibm.com

Abstract

Research in neural networks in the field of computer vision has achieved remarkable accuracy for point estimation. However, the uncertainty in the estimation is rarely addressed. Uncertainty quantification accompanied by point estimation can lead to a more informed decision, and even improve the prediction quality. In this work, we focus on uncertainty estimation in the domain of crowd counting. We propose a scalable neural network framework with quantification of decomposed uncertainty using a bootstrap ensemble. We demonstrate that the proposed uncertainty quantification method provides additional insight to the crowd counting problem and is simple to implement. We also show that our proposed method outperforms the current state of the art method in many benchmark datasets. To the best of our knowledge, we have the one of the best systems for ShanghaiTech part A and B, UCF-CC 50, UCSD, and the best for UCF-QNRF datasets.

1. Introduction

The counting problem is the estimation of the number of objects in a still image or video frame. It arises in many real-world applications including cell counting in microscopic images [63], monitoring crowds in surveillance systems [12], and counting the number of trees in an aerial image of a forest [16]. Especially in modern urban setting with increased deployments of cameras and surveillance systems, there is an increasing need for computational models which can analyze highly dense crowds using real time video feeds from surveillance cameras. Crowd counting is a crucial component of such an automated crowd analysis system. This involves estimating the number of people in the crowd, as well as the distribution of the crowd density over the entire area of the gathering. This is typically done in a supervised learning setting where annotated labels are provided.

Recently, convolutional neural network (CNN) has been shown to have successes in a wide range of tasks in computer vision, such as object detection [48], image recognition [17], face recognition [51] and image segmentation [35]. Inspired by these successes, many CNN based crowd counting methods have been proposed. Along with density estimation techniques [30], CNN based approaches have shown outstanding performances over previous works which were relying on handcrafted feature extraction. However, existing CNN based methods offer only point estimates of counts (or density map) and do not address the uncertainty in the prediction, which can come from the model and also from data itself. Probabilistic interpretations of outputs of the model via uncertainty quantification are important. When given a new unlabeled crowd image, how much can we trust the output of the model if it only provides a point estimate? Uncertainty quantification accompanied by point estimation can lead to a more informed decision, and even improve the prediction quality.

Uncertainty quantification is crucial also for the practitioners of these crowd counting methods. With the quantification of prediction confidence at hand one can treat uncertain inputs and special cases explicitly. For instance, a crowd counting model might return a density map (or count) with less confidence (high uncertainty) in some area of a given scene. In this case the practitioner could decide to pass the image – or the specific part of the image that the model is uncertain about – to a human for validation.

While Bayesian methods provide a mathematically plausible framework to deal with uncertainty quantification, often these methods come with a prohibitively computational cost. In this work, we propose a simple and scalable neural network framework using a bootstrap ensemble to quantify uncertainty for crowd counting. The key highlights of our work are:

- To the best of our knowledge, this work is the first to address uncertainty quantification of neural network predictions for crowd counting. Our method is shown to produce accurate estimated uncertainty.

- Our proposed method achieves state-of-the-art level performances on multiple crowd counting benchmark

datasets.

- Our proposed framework is generic and independent of the architecture of an underlying network. Combined with its simplicity for implementation, it can be easily adapted to another architecture.

2. Related Work

The previous literature on crowd counting problems can be categorized into three kinds of approaches depending on methodology: detection-based, regression-based and density-based methods.

Detection-based crowd counting is an approach to directly detect each of the target objects in a given image. A typical approach is to utilize object detectors [29, 31, 62] often using moving-windows [12]. Then, the counts of targets in an image is automatically given as a byproduct of detection results. These methods typically require well-trained classifiers to extract low-level features from the whole human body [11, 59]. However, objects can be highly occluded in many crowded scenes and many target objects can be in drastically different scales, making detection much more challenging. These issues make detection based approaches infeasible in dense crowd scenes.

Regression-based approaches [7, 9, 26, 49, 53] are proposed to remedy the occlusion problems which are obstacles for detection-based methods. Regression-based methods directly map input crowd images to scalar values of counts, hence bypassing explicit detection tasks. Particularly, a mapping between image features and the crowd count is learned. Typically the extracted features are used to generate low-level information, which is learned by a regression model. Hence, these methods leverage better feature extraction (if available) and regression algorithms for estimating counts [53, 1, 7, 9, 52]. For example, [6, 49, 8] take advantage of spatial or depth information and utilize segmentation methods to filter the background region and regress counts only on the foreground of images. However, these regression-based methods mostly ignore the spatial information in the crowd images.

Density-based crowd counting, originally proposed in [30], preserves both the count and spatial distribution of the crowd, and have been shown effective at object counting in crowd scenes. In an object density map, the integral over any sub-region is the number of objects within the corresponding region in the image. Density-based methods are generally better at handling cases where objects are severely occluded by bypassing the hard detection of every object, while also maintaining some spatial information about the crowd. [30] proposes a method which learns a linear mapping between the image feature and the density map. [45] proposes learning a non-linear mapping using random forest regression. However, earlier approaches still depended on hand-crafted features.

Density-based crowd counting using CNN. In recent years, the CNN based methods with density targets have shown performances superior to the traditional methods based on handcrafted features [13, 61, 64]. To address perspective issues, [66] leverages a multi-column network using convolution filters with different sizes in each column to generate the density map. As a different approach to address perspective issues, [43] proposes taking a pyramid of input patches into a network. [50] improves over [66] and uses a switching layer to classify the crowd into three classes depending on crowd density and to select one of 3 regressor networks for actual counting. [65] incorporates a multi-task objective, jointly estimating the density map and the total count by connecting fully convolutional networks and recurrent networks (LSTM). [57] uses global and local context to generate high quality density map. [32] introduces the dilated convolution to aggregate multi-scale contextual information and utilizes a much deeper architecture from VGG-16 [55]. [5] proposes an encoder-decoder network with the encoder extracting multi-scale features with scale aggregation modules and the decoder generating density maps by using a set of transposed convolutions.

Limitations of current state of the art: While density estimation and CNN based approaches have shown outstanding performances in the problems of crowd counting, less attention has been paid to assessing uncertainty in predictive outputs. Probabilistic interpretations via uncertainty quantification are important because (1) lack of understanding of model outputs may provide sub-optimal results and (2) neural networks are subject to over fitting, so making decisions based on point prediction alone may provide incorrect predictions with spuriously high confidence.

3. Uncertainty in Neural Networks

Much of the previous work on Bayesian neural network studied uncertainty quantification founded on parametric Bayesian inference [4, 14] (we defer a detailed discussion on Bayesian neural network to the appendix). In this work, we consider a non-parametric bootstrap of functions.

3.1. Bootstrap ensemble

Bootstrap is a simple technique for producing a distribution over functions with theoretical guarantees [3]. It is also general in terms of the class of models that we can accommodate. In its most common form, a bootstrap method takes as input a dataset $\mathcal{D}$ and a function f_θ. We can transform the original dataset $\mathcal{D}$ into K different datasets $\{\mathcal{D}_k\}_{k=1}^K$'s of cardinality equal to that of the original data $\mathcal{D}$ that is sampled uniformly with replacement. Then we train K different models. For each model f_{θ_k}, we train the model on the dataset $\mathcal{D}_k$. So each of these models is trained on data from the same distribution but on a different dataset. Then

if we want to approximate sampling from the distribution of functions, we sample uniformly an integer k from 1 to K and use the corresponding function f_{θ_k}.

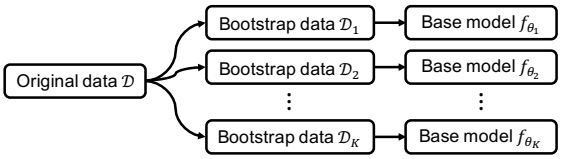

Figure 1. Bootstrap ensemble sampling. each base model is trained on randomly perturbed data

In cases of using neural networks as base models f_{θ_k}, bootstrap ensemble maintains a set of K neural networks $\{f_{\theta_k}\}_{k=1}^{K}$ independently on K different bootstrapped subsets of the data. It treats each network as independent samples from the weight distribution. In contrast to traditional Bayesian approaches discussed earlier, bootstrapping is a frequentist method, but with the use of the prior distribution, it could approximate the posterior in a simple manner. Also it scales nicely to high-dimensional spaces, since it only requires point estimates of the weights. However, one major drawback is that computational load increase linearly with respect to the number of base models. In the following section, we discuss how to mitigate this issue and still maintain a reasonable uncertainty estimates.

3.2. Measures of uncertainty

When we address uncertainty in predictive modeling, there are two major sources of uncertainty [22]:

1. **epistemic uncertainty** is uncertainty due to our lack of knowledge; we are uncertain because we lack understanding. In terms of machine learning, this corresponds to a situation where our model parameters are poorly determined due to a lack of data, so our posterior over parameters is broad.

2. **aleatoric uncertainty** is due to genuine stochasticity in the data. In this situation, an uncertain prediction is the best possible prediction. This corresponds to noisy data; no matter how much data the model has seen, if there is inherent noise then the best prediction possible may be a high entropy one .

Note that whether we apply a Bayesian neural network framework or a frequentist bootstrap ensemble framework, the kind of uncertainty which is addressed by either of the methods is epistemic uncertainty only. Epistemic uncertainty is often called as model uncertainty and it can be explained away given enough data (in theory as data size increases to infinity this uncertainty converges to zero). Addressing aleatoric uncertainty is also crucial for the crowd counting problem since many crowd images do possess inherent noise, occlusions, perspective distortions, etc. that regardless of how much data the model is trained on, there are certain aspects the model is not able to capture. Following [22], we incorporate both epistemic uncertainty and aleatoric uncertainty in a neural network for crowd counting. We discuss how we operationalize in a scalable manner in the following section.

3.3. Calibration of Predictive Uncertainty

Many Bayesian methods estimating predictive uncertainty often fail to capture the true distribution of the data [27]. For example, a 95% posterior confidence interval may not contain the true outcome 95% of the time. In such a case, the model is considered to be not *calibrated* [25]. Bootstrap ensemble methods we consider in this work are also not immune to this issue. Hence, we address this by incorporating a technique recently introduced in [25], which calibrates any regression methods including neural networks. The proposed procedure is inspired by Platt scaling [46] which recalibrates the predictions of a pre-trained classifier in a post-processing step. [25] show that the recalibration procedure applied to Bayesian models is guaranteed to produce calibrated uncertainty estimates given enough data. We discuss how we apply the recalibration procedure to our problem in more detail in Section 4.6.

4. Proposed Method

4.1. Single network with K output heads

Training and maintaining several independent neural networks is computationally expensive especially when each base network is a large and deep neural network. In order to remedy this issue, we adopt a single network framework which is scalable for generating bootstrap samples from a large and deep neural network [44]. The network consists of a shared architecture — for example, convolution layers — with K bootstrapped heads branching off independently. Each head is trained only on its bootstrapped sub-sample of the data as described in Section 3.1. The shared network learns a joint feature representation across all the data, which can provide significant computational advantages at the cost of lower diversity between heads. This type of bootstrap can be trained efficiently in a single forward/backward pass; it can be thought of as a data-dependent dropout, where the dropout mask for each head is fixed for each data point [58].

4.2. Capturing epistemic uncertainty

To capture epistemic uncertainty in a neural network, we put a prior distribution over its weights, for example a Gaussian prior: $[\theta_s, \theta_1, ..., \theta_K] \sim \mathcal{N}(0, \tilde{\sigma}^2)$, where θ_s is the parameter of the shared network and $\theta_1, ..., \theta_K$ are the parameters of bootstrap heads $1, .., K$. Let x be an image input and y be a density output. Without loss of generality, we

define our pixel-wise likelihood as a Gaussian with mean given by the model output: $p\left(y|f_\theta(x)\right) = \mathcal{N}\left(f_\theta(x), \sigma^2\right)$, with an observation noise variance σ^2.

For brevity of notations we overload the term $\theta_k = [\theta_k, \theta_s]$ since θ_s is shared across all samples. For each iteration of training procedure, we sample the model parameter $\hat{\theta}_k \sim q(\theta)$ where $q(\theta)$ is a bootstrap distribution. In other words, at each iteration we randomly choose which head to use to predict an output $\hat{y} = f_{\hat{\theta}_k}(x)$. Then the objective is to minimize the loss (for a single image x) given by the negative log-likelihood:

$$\mathcal{L}(\theta) = \frac{1}{D}\sum_i \frac{1}{2\sigma^2}\|y_i - \hat{y}_i\|^2 + \frac{1}{2}\log\sigma^2 \qquad (1)$$

where y_i is the i-th pixel of the output density y corresponding to input x and D is the number of output pixels. Note that the observation noise σ^2 which captures how much noise we have in the outputs stays constant for all data points. Hence we can further drop the second term (since it does not depend on θ), but for the sake of consistency with the following section where we discuss a heteroscedastic setting, we leave it as is. Now, epistemic uncertainty can be captured by the predictive variance, which can be approximated as:

$$\text{Var}(y) \approx \sigma^2 + \frac{1}{K}\sum_{k=1}^{K} f_{\hat{\theta}_k}(x)^\top f_{\hat{\theta}_k}(x) - \mathbb{E}(y)^\top\mathbb{E}(y) \qquad (2)$$

with approximated mean: $\mathbb{E}(y) \approx \frac{1}{K}\sum_{k=1}^{K} f_{\hat{\theta}_k}(x)$. Note that during training procedure we randomly select one output head but during test time we combine individual predictions from K heads to compute the predictive mean and the variance.

4.3. Incorporating aleatoric uncertainty

In contrast to homoscedastic settings where we assume the observation noise σ^2 is constant for all inputs, heteroscedastic regression assumes that σ^2 can vary with input x [28, 42]. This change can be useful in cases where parts of the observation space might have higher noise levels than others [22]. In crowd counting applications, it is often the case that images may come from different cameras and scenes. Also due to occlusion and perspective issues within a single image, it is often the case that observation noise can vary from one part of an image (or pixel) to another part (or pixel).

Following [22], the network outputs both the estimated density map y and the noise variance σ^2. Therefore, in our bootstrap implementation of the network, the output layer has a total of $K + 1$ nodes — K nodes corresponding to an ensemble of density map predictions y and an extra node corresponding to σ^2. Let θ_σ be the parameter corresponding to the output node of the noise variance σ^2. Now, as before, we overload the term $\theta_k = [\theta_k, \theta_s, \theta_\sigma]$ since θ_σ is shared across the bootstrap sampling. We draw a sample of model parameters from the approximate posterior given by bootstrap ensemble $\hat{\theta}_k \sim q(\theta)$. But this time as described above, we have two parallel outputs, the density map estimate $\hat{y}$ and the noise variance estimate $\hat{\sigma}^2$:

$$[\hat{y}, \hat{\sigma}^2] = f_{\hat{\theta}_k}(x).$$

Then, we have the following loss given input image x which we want to minimize:

$$\mathcal{L}(\theta) = \frac{1}{D}\sum_i \frac{1}{2\hat{\sigma}_i^2}\|y_i - \hat{y}_i\|^2 + \frac{1}{2}\log\hat{\sigma}_i^2. \qquad (3)$$

Note that this loss contains two parts: the least square residual term which depends on the model uncertainty (epistemic uncertainty) and an aleatoric uncertainty regularization term. Now, if the model predicts $\hat{\sigma}^2$ to be too high, then the residual term will not have much effect on updating the weights – the second term will dominate the loss. Hence, the model can learn to ignore the noisy data, but is penalized for that. In practice, due to the numerical stability of predicting σ^2 which should be positive, we predict the log variance $s_i := \log\sigma^2$ instead of σ^2 for the output [22].

4.4. Network architecture

First of all, note that our proposed framework is generic and is not restricted to a specific type of architecture. However, for the sake of concreteness and implementation, we use the architecture proposed in [32] (CSRNet) which has shown a state of art performance in crowd counting tasks. CSRNet extends the VGG-16 [55] with the dilated convolution achieving the top performance of the state of the art in crowd counting. For discussion on dilated convolution, we refer the readers to [32]. The network is composed of two major components: a CNN as the front-end for feature extraction and a dilated CNN for the back-end, which uses dilated kernels to deliver larger reception fields and to replace pooling operations. We replace the output layer with the K bootstrap ensemble heads for $\hat{y}$ and another output for $\hat{\sigma^2}$. We call our network DUB-CSRNet where "DUB" stands for decomposed uncertainty using bootstrap. The details of the architecture is shown in Figure 2.

4.5. Training procedure

We initialize the front-end layers (the first 10 convolutional layers) in our model with the corresponding part of a pre-trained VGG-16 [55]. For the rest of the parameters, we initialize with a Gaussian distribution with mean 0 and standard deviation 0.01. Given a training dataset of input images $X = \{x_1, ..., x_N\}$ and corresponding ground truth density maps $Y = \{y_1, ..., y_N\}$, at each iteration, we sample uniformly at random $k \in \{1, ..., K\}$ to choose an output head k and predict $[\hat{y}_n, \hat{s}_n] = f_{\hat{\theta}_k}(x_n)$ for n-th image

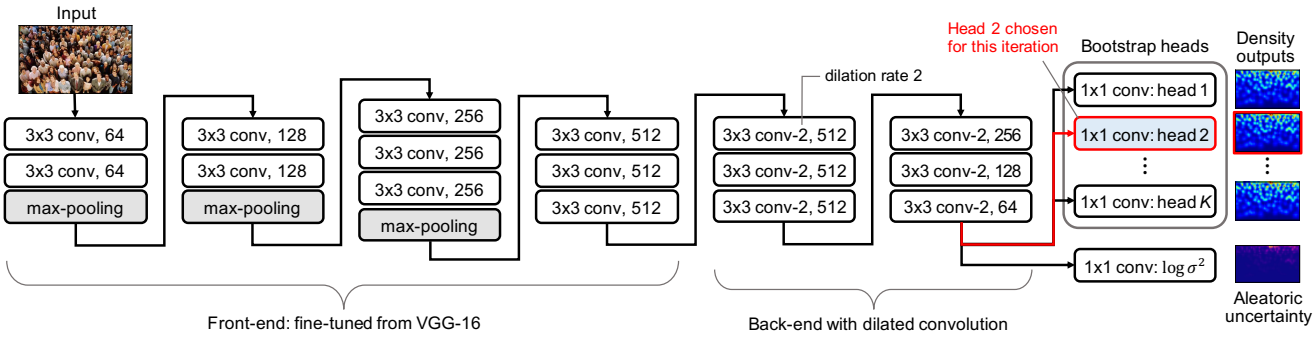

Figure 2. Network architecture of our proposed method, DUB-CSRNet. All convolutional layers use SAME padding to maintain the size of the output the same as the input size. Max-pooling layers are applied with 2×2 window with stride size 2. The back-end layers use dilated kernels with rate 2. The output layer branches out to K bootstrap heads and an extra log-variance output.

as discussed in the previous sections. Algorithm 1 presents a single-image batch training procedure. $\hat{y}_{n,i}$ and $\hat{s}_{n,i}$ are the i-th pixel of the estimated density map and the log variance respectively corresponding to input image x_n. D_n is the number of output pixels of y_n. Note that due to pooling operations, the number of output pixels is the same as the number of input pixels. Adam optimizer [23] with a learning rate of 10^{-5} is applied to train the model.

Algorithm 1 Decomposed Uncertainty using Bootstrap

Require: Input images $\{x_n\}_{n=1}^N$, GT density $\{y_n\}_{n=1}^N$

Initialize parameters θ
for each epoch **do**
  for all $n = 1$ to N **do**
    Sample a bootstrap head $k \sim$ Uniform$\{1, ..., K\}$
    Compute predictions $[\hat{y}_n, \hat{s}_n] = f_{\hat{\theta}_k}(x_n)$
    Compute loss:

$$\mathcal{L}(\theta_k) = \frac{1}{D_n} \sum_i \frac{1}{2\exp(\hat{s}_{n,i})} \|y_{n,i} - \hat{y}_{n,i}\|^2 + \frac{1}{2}\hat{s}_{n,i}$$

    Update θ_k using gradient $\frac{d\mathcal{L}(\theta_k)}{d\theta_k}$
  end for
end for

4.6. Recalibration of Predictive Uncertainty

Once we have a trained model $f_{\hat{\theta}}$, we compute the mean prediction $\mu(x_n) = \frac{1}{K}\sum_k f_{\hat{\theta}_k}(x_n)$ for an input image x_n. Note that $\mu(x_n)$ is a density map. We sum over all pixels in $\mu(x_n)$ to compute the predicted mean count $\bar{C}_n$. Similarly, using the (pixel-wise) predictive variance in Eq.(2), we compute the predictive standard deviation in counts $\bar{\sigma}_n$ by summing over all pixels. Then we construct a standardized residual $Z_n = (C_n - \bar{C}_n)/\bar{\sigma}_n$ where C_n is the ground-truth count for image x_n and construct a quantile target $\hat{P}(Z_n)$ which is the proportion of data whose standardized residual is below Z_n. Then using each pair $(Z_n, \hat{P}(Z_n))$, we fit an isotonic regression model $\mathcal{R}$. The recalibration procedure is summarized in Algorithm 2.

Algorithm 2 Uncertainty Recalibration

Require: $\{C_n, \bar{C}_n, \bar{\sigma}\}_{n=1}^N$ for validation data

Compute $Z_n = (C_n - \bar{C}_n)/\bar{\sigma}_n$ for all n
Construct a recalibration dataset:

$$\tilde{\mathcal{D}} = \left\{ \left(Z_n, \hat{P}(Z_n) \right) \right\}_{n=1}^N$$

where $\hat{P}(z) = |\{C_m \mid Z_m \leq z, m = 1, ..., N\}|/N$
Train a isotonic regression model $\mathcal{R}$ on $\tilde{\mathcal{D}}$.

Note that the recalibration dataset $\tilde{\mathcal{D}}$ is constructed using a validation data (non-training data) and the model $\mathcal{R}$ is fitted on this dataset. Once $\mathcal{R}$ is learned, for a given quantile p, (e.g. 0.95 and 0.05 for 90% confidence interval) one can easily find $Z^p \in \mathbb{R}$ such that $\mathcal{R}(Z^p) \approx p$ (since $\mathcal{R}$ is a monotone function). When using at a test time where we only have $\bar{C}_n$ and $\bar{\sigma}_n$, we can construct a confidence bound by computing $\bar{C}_n + \bar{\sigma}_n Z^p$

5. Experiments

In this section, we first introduce datasets and experiment details. We give the evaluation results and perform comparisons between the proposed method with recent state-of-the-art methods. For all experiments, we used $K = 10$ heads for DUB-CSRNet. We follow the standard procedure to generate the ground truth density map using Gaussian kernels (we defer the details to the appendix).

5.1. Evaluation metrics

For crowd counting evaluation, the count estimation error is measured by two metrics, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), which are commonly used for quantitative comparison in previous works. They are defined as follows:

$$\text{MAE} = \frac{1}{N}\sum_{n=1}^{N}|\hat{C}_n - C_n|, \quad \text{RMSE} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(\hat{C}_n - C_n)^2}$$

where N is the number of test samples, C_n is the true crowd count for the n-th image sample and $\hat{C}_n$ is the corresponding estimated count. C_n and $\hat{C}_n$ are given by the integration over the ground truth density map $\sum_i y_{n,i}$ and over an estimated density map $\sum_i \hat{y}_{n,i}$ respectively, where i is the i-th pixel in output images. Note that during test time we use predictive mean over K bootstrap outputs as $\hat{y}$.

5.2. Ablation study

We performed ablation studies on UCF-CC 50 and UCF-QNRF datasets to validate the efficacy of our proposed method. We first compared with two variants where we incorporate either aleatoric uncertainty or epistemic uncertainty only. "Epistemic uncertainty only" model refers to a bootstrap ensemble model with a minimization loss defined as Eq.(1) where σ^2 is fixed for all inputs. "Aleatoric uncertainty only" model is a single neural network without bootstrap but using a heteroscedastic observation noise as in Eq.(3). We include CSRNet as the base model. Also, in order to test the adaptability of our framework to other architecture, we applied the DUB extension to MCNN [66] with branching outputs and an aleatoric uncertainty output. We compare with the standard MCNN. The results in Table 1 show that our proposed framework combining both aleatoric and epistemic uncertainty contributes significantly to performance on the evaluation data, and can be applied to other architecture.

Methods	UCF-CC 50 MAE	UCF-QNRF MAE
CSRNet [32]	266.1	135.5
CSRNet + aleatoric only	261.7	132.1
CSRNet + epistemic only	249.2	124.8
DUB-CSRNet (Ours)	**235.2**	**116.3**
MCNN [66]	377.6	277.0
MCNN + **DUB** extension	**359.4**	**254.6**

Table 1. Ablation study on uncertainty components

5.3. Performance comparisons

We evaluate our method on four publicly available crowd counting datasets: ShanghaiTech [66], UCF-CC 50 [20], UCSD [6], and UCF-QNRF [21]. For all datasets, we generate ground truth density maps with fixed spread Gaussian kernel (see Section C in the appendix for details). We compare our method with previously published work. In each table, the previous work which provided code or have been validated by a third party other than the original authors have been listed above our method. For completeness, we also list the recent work (without code or validation by a third party) below our method and include the numbers reported by the original authors. We highlight the best two performances in each metric.

Method	Part A MAE	Part A RMSE	Part B MAE	Part B RMSE
Zhang et al. [64]	181.3	277.7	32.0	49.8
Marsden et al. [40]	126.5	173.5	23.8	33.1
MCNN [66]	110.2	173.2	26.4	41.3
Cascaded-MTL [56]	101.3	152.4	20.0	31.1
Switch-CNN [50]	90.4	135.0	21.6	33.4
CP-CNN [57]	73.6	**106.4**	20.1	30.1
D-ConvNet [54]	73.5	112.3	18.7	26.0
L2R [34]	73.6	112.0	13.7	21.4
CSRNet [32]	68.2	115.0	10.6	16.0
DUB-CSRNet (Ours)	**66.4**	111.1	**9.4**	**15.1**
DRSAN [33]	69.3	**96.4**	11.1	18.2
ic-CNN [47]	68.5	116.2	10.7	16.0
SANet [5]	**67.0**	104.5	**8.4**	**13.6**

Table 2. Estimation errors on ShanghaiTech dataset

5.3.1 Datasets and experiment setup

ShanghaiTech. The ShanghaiTech dataset [66] contains 1198 annotated images with a total of 330,165 persons. This dataset consists of two parts: Part A which contains 482 images and Part B which contains 716 images. Part A is randomly collected from the Internet and contains mostly highly congested scenes. Part B contains images captured from street views with relatively sparse crowd scenes. We use the training and testing splits provided by the authors: 300 images for training and 182 images for testing in Part A; 400 images for training and 316 images for testing in Part B. Table 2 presents the evaluation results of our method compared to other previous works.

UCF-CC 50. The UCF-CC 50 dataset [64] is a small dataset which contains only 50 annotated crowd images. However, the challenging aspect of this dataset is that there is a large variation in crowd counts which range from 94 to 4543. Along with this variation, the limited number of images makes it a challenging dataset for the crowd counting tasks. Since training and test data split is not provided, as done in the previous literature [32, 64], We use 5-fold cross-validation to evaluate the performance of the proposed method. The results are shown in Table 3.

UCSD. The UCSD dataset [6] consists of 2000 frames captured by surveillance cameras. The images contain low density crowds ranging from 11 to 46 persons per image. The region of interest (ROI) is provided with the data to eliminate irrelevant objects in the images. We process the annotations with ROI. The low resolution of the images

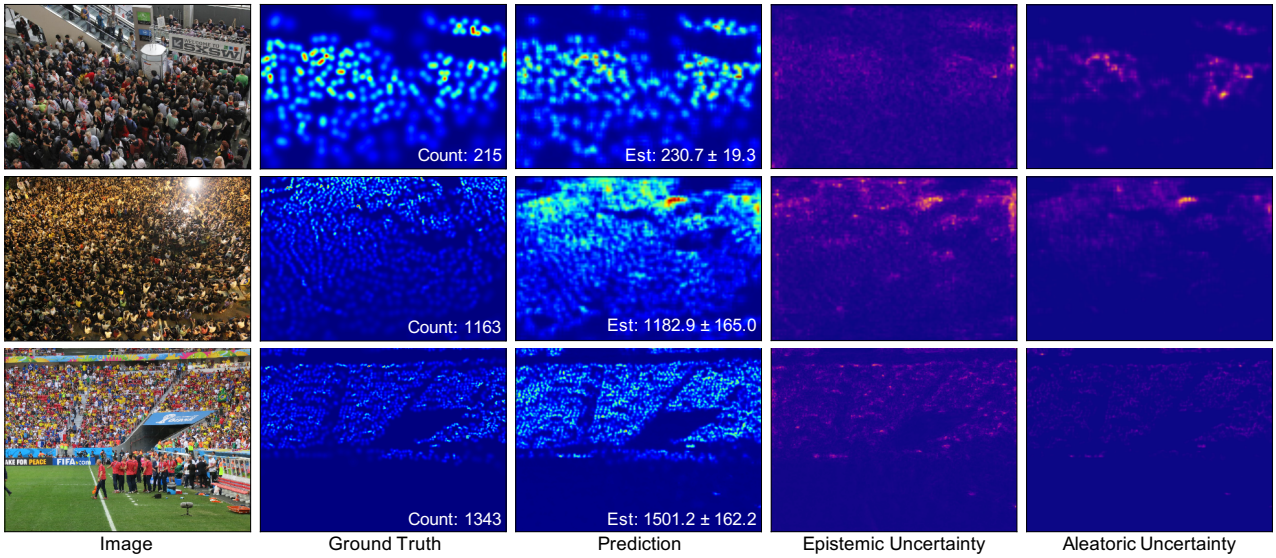

Figure 3. Qualitative results of DUB-CSRNet on the ShanghaiTech dataset and the UCF-QNRF dataset. For each image, we demonstrate the ground truth density maps and counts, the estimated density maps and estimated counts with 90% confidence interval. We also present both estimated epistemic and aleatoric uncertainty quantification. More red color means higher uncertainty. Epistemic uncertainty captures the model's lack of knowledge on the data. Aleatoric uncertainty captures inherent noise in the data.

Method	MAE	RMSE
Idrees et al. [19]	419.5	541.6
Zhang et al. [64]	467.0	498.5
MCNN [66]	377.6	509.1
Onoro et al. [43] Hydra-2s	333.7	425.2
Walach et al. [60]	364.4	341.4
Marsden et al. [40]	338.6	424.5
Cascaded-MTL [56]	322.8	397.9
Switch-CNN [50]	318.1	439.2
CP-CNN [57]	295.8	**320.9**
D-ConvNet [54]	288.4	404.7
L2R [34]	279.6	388.9
CSRNet [32]	266.1	397.5
DUB-CSRNet (Ours)	**235.2**	332.7
DRSAN [33]	**219.2**	**250.2**
ic-CNN [47]	260.9	365.5
SANet [5]	258.4	334.9

Table 3. Estimation errors on UCF-CC 50 dataset

(238×158) makes it challenging to generate density maps especially with the use of pooling operations. So we perform up-sampling of the images following [32]. MAE and RMSE are evaluated only in the specified ROI during testing. We use frames 601 through 1400 as training set and the rest of the frames as testing set following [6]. The evaluation results are shown in Table 4.

UCF-QNRF. The UCF-QNRF dataset was recently introduced by [21]. It is currently the largest crowd dataset

Method	MAE	RMSE
Zhang et al. [64]	1.60	3.31
CCNN [43]	1.51	-
Switch-CNN [50]	1.62	2.10
FCN-rLSTM [65]	1.54	3.02
CSRNet [32]	1.16	1.47
MCNN [66]	1.07	1.35
DUB-CSRNet (Ours)	**1.03**	**1.24**
SANet [5]	**1.02**	**1.29**

Table 4. Estimation errors on UCSD dataset

which contains 1,535 images with dense crowds with many of them being high resolution images. Approximately 1.25 million people were annotated with dot annotations These images come with a wider variety of scenes and contains the most diverse set of viewpoints, densities, and lighting variations. The ground truth counts of the images in the dataset range from 49 to 12,865. Meanwhile, the median and the mean counts are 425 and 815.4, respectively. The training dataset contains 1,201 images, with which we train our model. Some of the images are so high-resolution that we faced memory issues in GPU while training. Hence, we down-sampled images that contains more than 3 million pixels. Then, we test our model on the remaining 334 images in the test dataset. The results are shown in Table 5.

Method	MAE	RMSE
Idrees et al.(2013) [19]	315	508
MCNN [66]	277	426
Encoder-Decoder [2]	270	478
CTML [56]	252	514
Switch-CNN [50]	228	445
Resnet101 [17]	190	277
Densenet201 [18]	163	226
Idrees et al.(2018) [21]	**132**	**191**
DUB-CSRNet (Ours)	**116**	**178**

Table 5. Estimation errors on UCF-QNRF dataset

5.3.2 Results

The results in Tables 2, 3, 4, and 5 show that our proposed method is within the top two performers for all of the benchmark datasets we consider. Comparing with the methods with publicly available code or validation by a third party, our method achieves the lowest MAE (the highest count accuracy) across all datasets.

5.4. Estimated uncertainty validation

Estimated uncertainty is meaningful if it can capture the true distribution of the data. As mentioned earlier in the paper, we can validate whether estimated uncertainty is well calibrated or not by checking whether the estimated p quantile confidence interval (CI) contains the true outcome p fraction of the time. Table 6 shows the fraction of test data in each dataset whose ground truth falls in 90% CI.[1] The results suggest that our estimated uncertainty is accurate.

Dataset	Ground truth in 90% CI
ShanghaiTech A	0.907
ShanghaiTech B	0.915
UCSD	0.911
UCF-QNRF	0.890

Table 6. Fraction of test data whose ground truth falls in 90% CI

5.5. Discussion on estimated uncertainty

Figure 3 visualize the samples along with estimated density maps and their epistemic and aleatoric uncertainty from test evaluations on the ShanghaiTech data and the UCF-QNRF data. The results demonstrate that the model is generally less confident (i.e. higher epistemic uncertainty) in dense crowd regions of the images, which is natural. There appears to be a certain level of positive correlation between epistemic and aleatoric uncertainty which is expected – since the common issues in crowd images such as occlusion and perspective issues are typically correlated with higher crowd density, this can cause both epistemic and aleatoric uncertainty to be higher. But, we also observe a notable difference in the estimated measures of uncertainty in the samples. We observe that aleatoric uncertainty is more prominent in areas where the image itself has more noise (for example, lighting glare in the second image in Figure 3) and occlusions (right side along the horizontal center line in the first image in Figure 3). We can observe that even in very crowded scenes, when occlusions and noise are less prominent, the estimated aleatoric uncertainty can be low – for example, the stadium image (the third image in Figure 3) shows very low aleatoric uncertainty over the entire image since there are rarely occlusions or perspective issues due to the stadium seating configuration.

[1] UCF-CC 50 dataset is not included since the uncertainty recalibration is difficult to perform due to the limited data size.

5.6. Prediction on real world data

Earlier in the paper, we raised a question of how much we can trust predictions of a model, especially when we do not have labels or ground truth to verify the accuracy of the predictions. Now with uncertainty estimates at hand, we can present crowd counting predictions on new real world data. In the supplementary material, we show our results on CNN's[2] giga-pixel images [10] which contains ultra high resolution (64,000 $\times$ 64,000 pixels) crowd images.

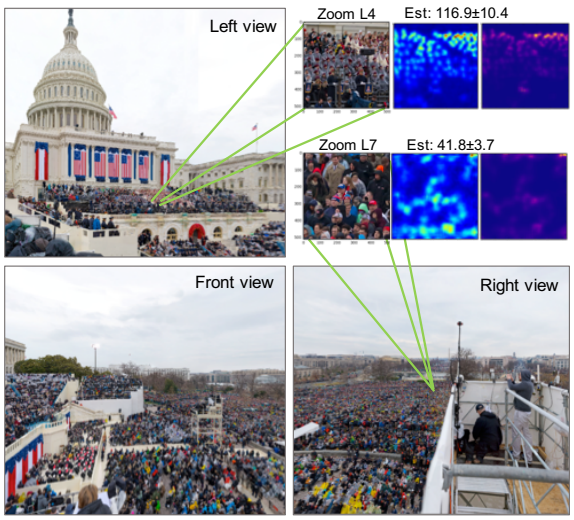

Figure 4. Snapshots of prediction on the giga-pixel images of the 2017 U.S. presidential inauguration

6. Conclusion

In this paper, we present a scalable and effective single neural network framework which can incorporate uncertainty quantification in prediction for crowd counting. The main component of the framework is combining shared con-

[2] Cable News Network

volutional layers and bootstrap ensembles to quantify uncertainty which is decomposed into epistemic and aleatoric uncertainty. Our proposed framework is generic, independent of the architecture choices, and also easily adaptable to other CNN based crowd counting methods. The extensive experiments demonstrate that the proposed method, DUB-CSRNet, has the state-of-the-art level performance on all benchmark datasets considered, and produces calibrated and meaningful uncertainty estimates.

References

[1] C. Arteta, V. Lempitsky, J. A. Noble, and A. Zisserman. Interactive object counting. In *European Conference on Computer Vision*, pages 504–518. Springer, 2014. 2

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015. 8

[3] P. J. Bickel and D. A. Freedman. Some asymptotic theory for the bootstrap. *The Annals of Statistics*, pages 1196–1217, 1981. 2

[4] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015. 2, 18

[5] X. Cao, Z. Wang, Y. Zhao, and F. Su. Scale aggregation network for accurate and efficient crowd counting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 734–750, 2018. 2, 6, 7

[6] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7. IEEE, 2008. 2, 6, 7

[7] A. B. Chan and N. Vasconcelos. Bayesian poisson regression for crowd counting. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 545–551. IEEE, 2009. 2

[8] A. B. Chan and N. Vasconcelos. Counting people with low-level features and bayesian regression. *IEEE Transactions on Image Processing*, 21(4):2160–2177, 2012. 2

[9] K. Chen, C. C. Loy, S. Gong, and T. Xiang. Feature mining for localised crowd counting. In *BMVC*, volume 1, page 3, 2012. 2

[10] CNN.com. Gigapixel: the Inauguration of Donald Trump. https://edition.cnn.com/interactive/2017/01/politics/trump-inauguration-gigapixel/. Accessed: 2018-11-15. 8, 13

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005. 2

[12] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):743–761, 2012. 1, 2

[13] M. Fu, P. Xu, X. Li, Q. Liu, M. Ye, and C. Zhu. Fast crowd density estimation with convolutional neural networks. *Engineering Applications of Artificial Intelligence*, 43:81–88, 2015. 2

[14] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016. 2, 18

[15] A. Graves. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356, 2011. 18

[16] O. Hassaan, A. K. Nasir, H. Roth, and M. F. Khan. Precision forestry: trees counting in urban areas using visible imagery based on an unmanned aerial vehicle. *IFAC-PapersOnLine*, 49(16):16–21, 2016. 1

[17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 8

[18] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017. 8

[19] H. Idrees, I. Saleemi, C. Seibert, and M. Shah. Multi-source multi-scale counting in extremely dense crowd images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2547–2554, 2013. 7, 8

[20] H. Idrees, K. Soomro, and M. Shah. Detecting humans in dense crowds using locally-consistent scale prior and global occlusion reasoning. *IEEE transactions on pattern analysis and machine intelligence*, 37(10):1986–1998, 2015. 6

[21] H. Idrees, M. Tayyab, K. Athrey, D. Zhang, S. Al-Maadeed, N. Rajpoot, and M. Shah. Composition loss for counting, density map estimation and localization in dense crowds. *arXiv preprint arXiv:1808.01050*, 2018. 6, 7, 8

[22] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017. 3, 4

[23] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[24] D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pages 2575–2583, 2015. 18

[25] V. Kuleshov, N. Fenner, and S. Ermon. Accurate uncertainties for deep learning using calibrated regression. In *International Conference on Machine Learning*, pages 2801–2809, 2018. 3

[26] S. Kumagai, K. Hotta, and T. Kurita. Mixture of counting CNNs: Adaptive integration of CNNs specialized to specific appearance for crowd counting. *arXiv preprint arXiv:1703.09393*, 2017. 2

[27] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017. 3

[28] Q. V. Le, A. J. Smola, and S. Canu. Heteroscedastic gaussian process regression. In *Proceedings of the 22nd international conference on Machine learning*, pages 489–496. ACM, 2005. 4

[29] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *null*, pages 878–885. IEEE, 2005. 2

[30] V. Lempitsky and A. Zisserman. Learning to count objects in images. In *Advances in neural information processing systems*, pages 1324–1332, 2010. 1, 2

[31] M. Li, Z. Zhang, K. Huang, and T. Tan. Estimating the number of people in crowded scenes by mid based foreground segmentation and head-shoulder detection. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE, 2008. 2

[32] Y. Li, X. Zhang, and D. Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 4, 6, 7

[33] L. Liu, H. Wang, G. Li, W. Ouyang, and L. Lin. Crowd counting using deep recurrent spatial-aware network. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, pages 849–855. AAAI Press, 2018. 6, 7

[34] X. Liu, J. van de Weijer, and A. D. Bagdanov. Leveraging unlabeled data for crowd counting by learning to rank. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7661–7669, 2018. 6, 7

[35] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1

[36] C. Louizos and M. Welling. Structured and efficient variational deep learning with matrix gaussian posteriors. In *International Conference on Machine Learning*, pages 1708–1716, 2016. 18

[37] C. Louizos and M. Welling. Multiplicative normalizing flows for variational bayesian neural networks. *arXiv preprint arXiv:1703.01961*, 2017. 18

[38] D. J. MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992. 18

[39] S.-i. Maeda. A bayesian encourages dropout. *arXiv preprint arXiv:1412.7003*, 2014. 18

[40] M. Marsden, K. McGuinness, S. Little, and N. E. O'Connor. Fully convolutional crowd counting on highly congested scenes. *arXiv preprint arXiv:1612.00220*, 2016. 6, 7

[41] R. M. Neal. Bayesian learning via stochastic dynamics. In *Advances in neural information processing systems*, pages 475–482, 1993. 18

[42] D. A. Nix and A. S. Weigend. Estimating the mean and variance of the target probability distribution. In *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference On*, volume 1, pages 55–60. IEEE, 1994. 4

[43] D. Onoro-Rubio and R. J. López-Sastre. Towards perspective-free object counting with deep learning. In *European Conference on Computer Vision*, pages 615–629. Springer, 2016. 2, 7

[44] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy. Deep exploration via bootstrapped dqn. In *Advances in neural information processing systems*, pages 4026–4034, 2016. 3

[45] V.-Q. Pham, T. Kozakaya, O. Yamaguchi, and R. Okada. Count forest: Co-voting uncertain number of targets using random forest for crowd density estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3253–3261, 2015. 2

[46] J. Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999. 3

[47] V. Ranjan, H. Le, and M. Hoai. Iterative crowd counting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 270–285, 2018. 6, 7

[48] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 1

[49] D. Ryan, S. Denman, C. Fookes, and S. Sridharan. Crowd counting using multiple local features. In *Digital Image Computing: Techniques and Applications, 2009. DICTA'09.*, pages 81–88. IEEE, 2009. 2

[50] D. B. Sam, S. Surya, and R. V. Babu. Switching convolutional neural network for crowd counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 6, 2017. 2, 6, 7, 8

[51] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015. 1

[52] S. Seguí, O. Pujol, and J. Vitria. Learning to count with deep object features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 90–96, 2015. 2

[53] C. Shang, H. Ai, and B. Bai. End-to-end crowd counting via joint learning local and global count. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 1215–1219. IEEE, 2016. 2

[54] Z. Shi, L. Zhang, Y. Liu, X. Cao, Y. Ye, M.-M. Cheng, and G. Zheng. Crowd counting with deep negative correlation learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5382–5390, 2018. 6, 7

[55] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2, 4

[56] V. A. Sindagi and V. M. Patel. CNN-based cascaded multi-task learning of high-level prior and density estimation for crowd counting. In *Advanced Video and Signal Based Surveillance (AVSS), 2017 14th IEEE International Conference on*, pages 1–6. IEEE, 2017. 6, 7, 8

[57] V. A. Sindagi and V. M. Patel. Generating high-quality crowd density maps using contextual pyramid CNNs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, page 18611870, 2017. 2, 6, 7

[58] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 3, 18

[59] P. Viola and M. J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004. 2

[60] E. Walach and L. Wolf. Learning to count with CNN boosting. In *European Conference on Computer Vision*, pages 660–676. Springer, 2016. 7

[61] C. Wang, H. Zhang, L. Yang, S. Liu, and X. Cao. Deep people counting in extremely dense crowds. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 1299–1302. ACM, 2015. 2

[62] L. Wang and N. H. Yung. Crowd counting and segmentation in visual surveillance. In *Image Processing (ICIP), 2009*

16th IEEE International Conference on, pages 2573–2576. IEEE, 2009. 2

[63] W. Xie, J. A. Noble, and A. Zisserman. Microscopy cell counting and detection with fully convolutional regression networks. *Computer methods in biomechanics and biomedical engineering: Imaging & Visualization*, 6(3):283–292, 2018. 1

[64] C. Zhang, H. Li, X. Wang, and X. Yang. Cross-scene crowd counting via deep convolutional neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 833–841. IEEE, 2015. 2, 6, 7

[65] S. Zhang, G. Wu, J. P. Costeira, and J. M. Moura. FCN-RLSTM: Deep spatio-temporal neural networks for vehicle counting in city cameras. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 3687–3696. IEEE, 2017. 2, 7

[66] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma. 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, 2016. 2, 6, 7, 8, 18

Supplementary Material for Crowd Counting with Decomposed Uncertainty

A. CNN giga-pixel imagery for the 2017 U.S. Presidential inauguration

Cable News Network (CNN) released the giga-pixel images of the 2017 U.S. Presidential inauguration [10]. The CNN giga-pixel images consist of the photos taken from 6 different viewing directions: left, front, right, up, back and down views as shown in Figure 5. Each viewing direction contains 15,625 (125×125) patches of 500×500 pixel images, which means that each viewing direction contains a total of approximately 3.9×10^9 pixels (3.9 giga-pixels). Additionally, the zoomed out versions (with 7 different zoom levels) of the images are also included. However, in fact they are stitched images of the original high resolution images (not additional photos).

To the best of our knowledge, this is the largest high resolution crowd image dataset from a single event. We present crowd counting prediction results on these giga-pixel images using our proposed method, DUB-CSRNet. Clearly, the dataset does not contain the ground truth labels or aggregated counts. Hence, merely performing a point estimation of counts (or density maps) on these images is not sufficient. In the evaluations on the benchmark datasets we presented in the paper, DUB-CSRNet not only demonstrates the state of art performance in terms of counting accuracy but also presents an insightful uncertainty measures for its prediction. With this uncertainty estimates at hand, we can perform predictions and measure how confident we are in our predictions.

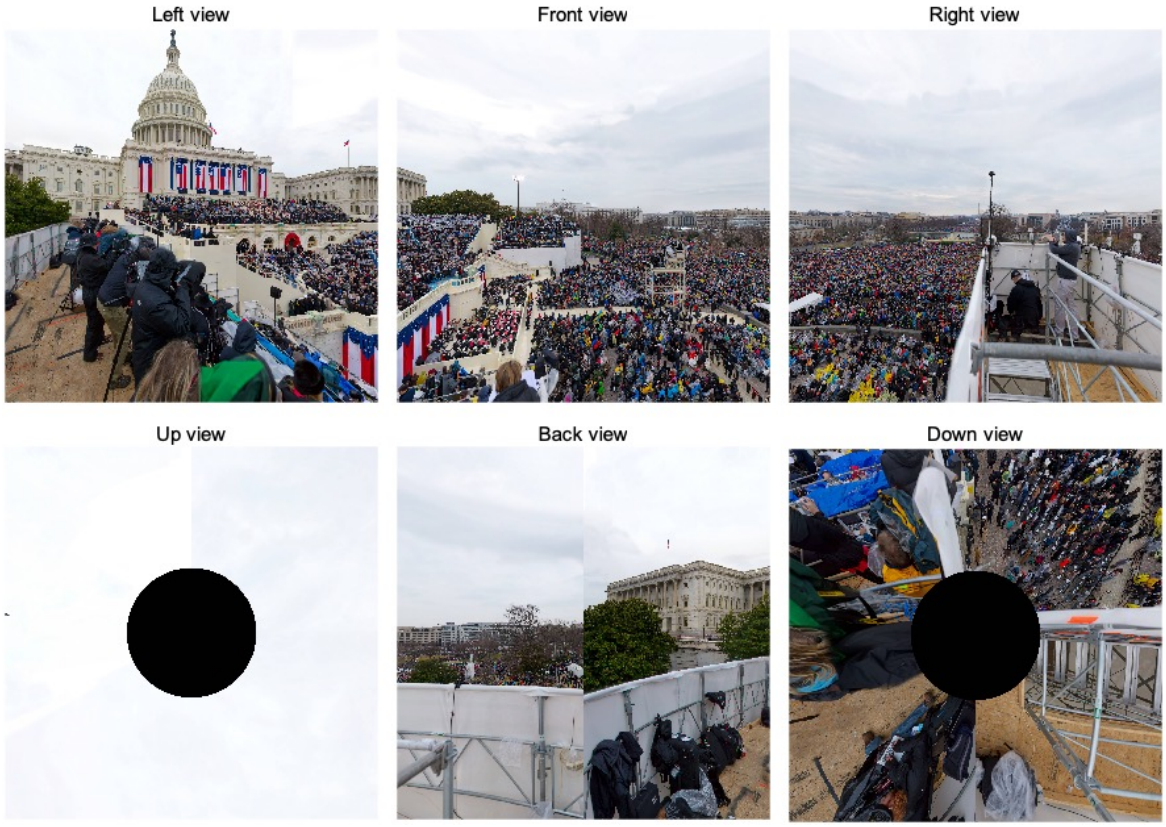

Figure 5. Different viewing directions of the CNN giga-pixel images

We observe that the majority of the crowd are captured in the three views: left, front and fright views. Hence, we present the results from these three views. We use a network that was trained on the UCF-QNRF dataset for prediction.

Note that while the method attempts to predict the count of the people present in the images, this is "not" about estimating a total number of attendees at the inauguration. We acknowledge that even if we could count accurately every person present in the image, that estimated number of people would be still smaller than the actual number of people who attended the inauguration since there are parts of the event that the giga-pixel images are not able to capture. For example, in Figure 7, a significant portion of the crowd is blocked from the view by the white temporary structure on which the camera operators are standing. We do not attempt any inference on what is not seen in the images.

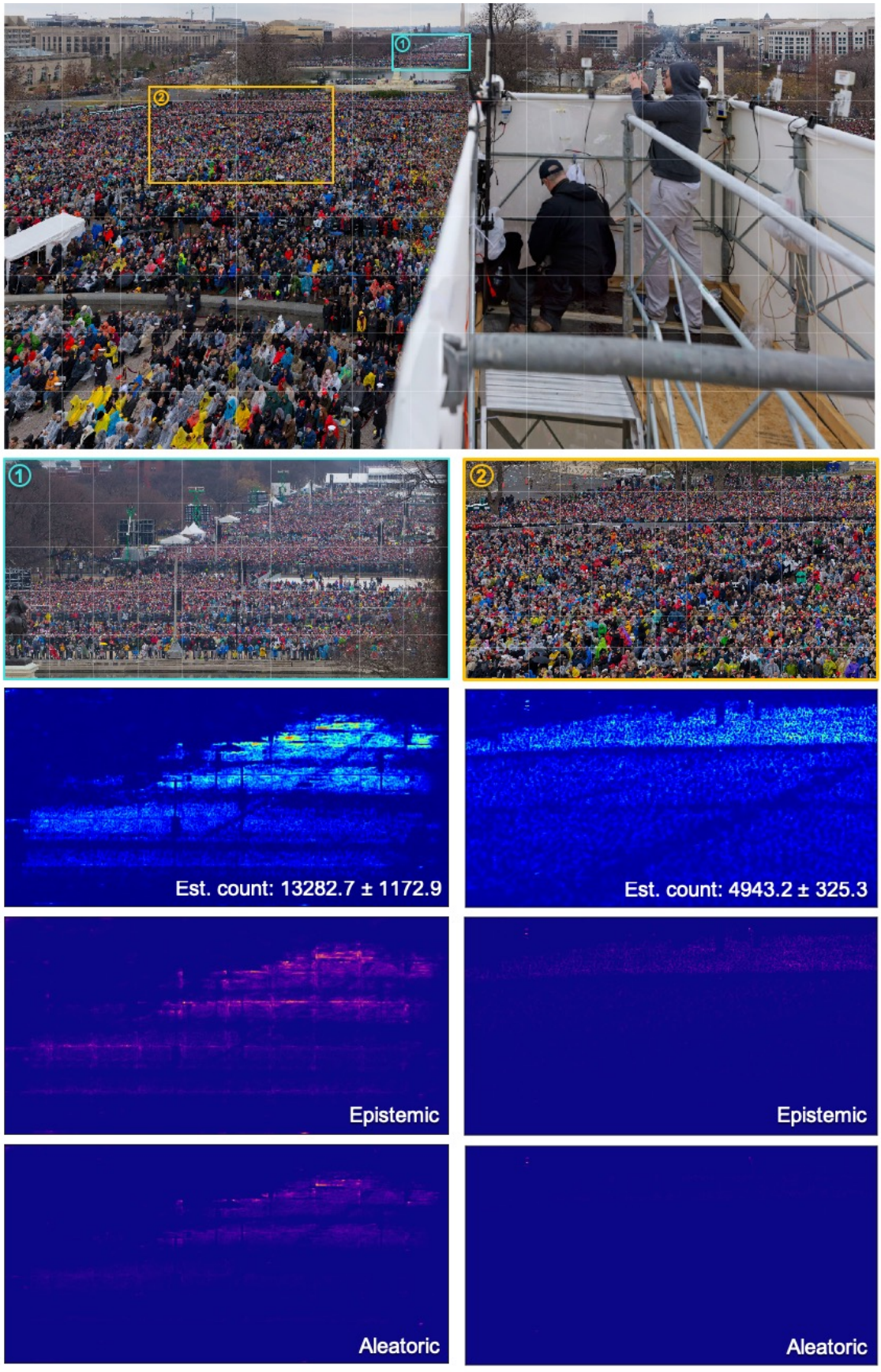

Figure 6. The right view of the CNN giga-pixel images and prediction examples

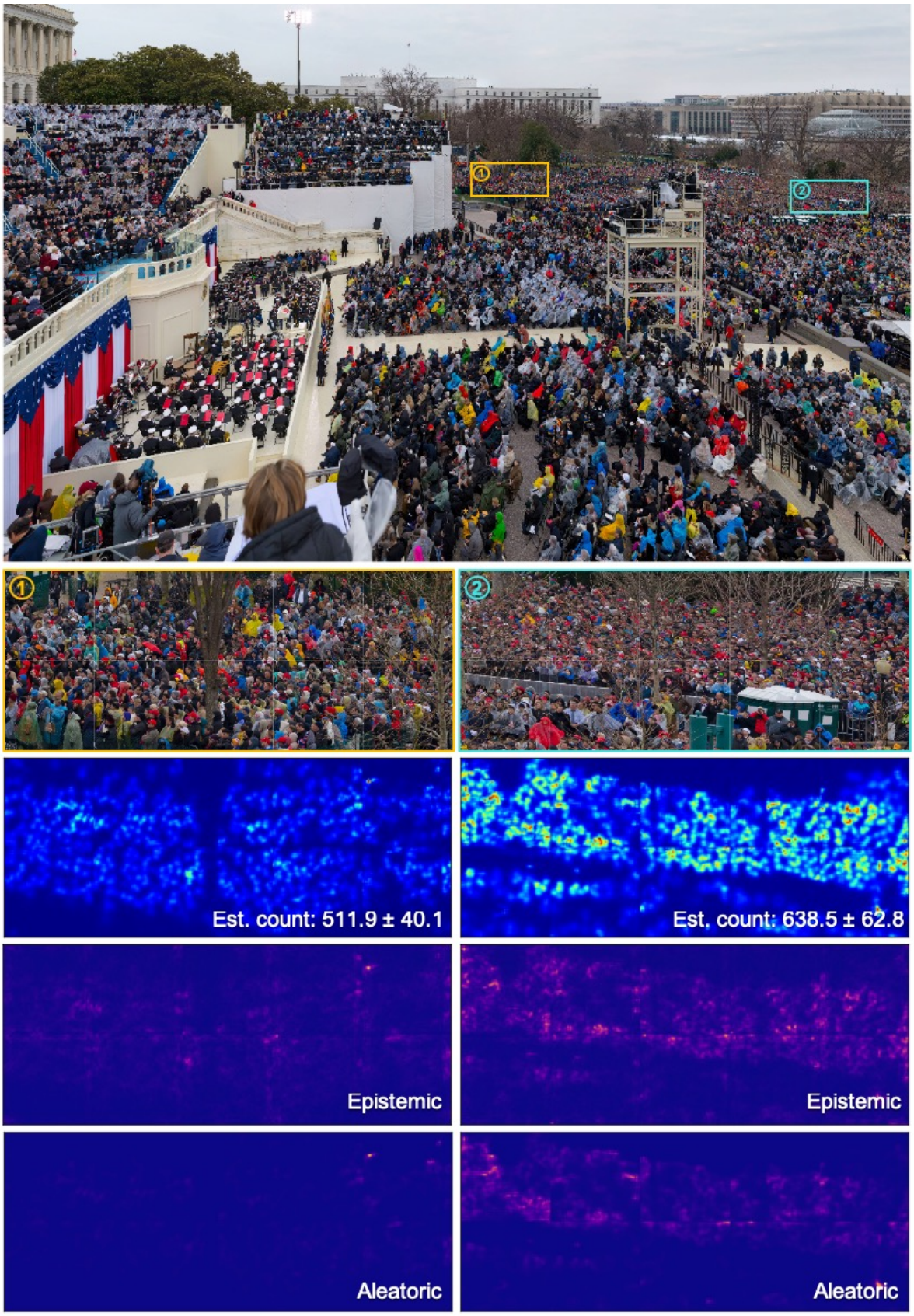

Figure 7. The front view of the CNN giga-pixel images and prediction examples

Figure 8. The left view of the CNN giga-pixel images and prediction examples

While one could simply apply DUB-CSRNet (or any other neural network based crowd counting method) directly to the giga-pixel images with each single direction image collage being a one large image input, it would not be a good practice. That is because it is highly likely to encounter a memory issue due to the large size and also because the trained network has not seen a person in such a larger scale (for example, consider the pixel sizes of cameramen on the bottom left region of Figure 8). One can pass each image with a fixed zoom level as input to the network. The results using a different zoom level (but fixed throughout prediction) are shown in Table 7. The estimated counts are provided with the 90% confidence interval. Note that since the ground-truth density or counts do not exist, we used the calibration method validated on UCF-QNRF test dataset.[3]

Zoom	Left view Est. count	Front view Est. count	Right view Est. count	Total
Level 3	1121.5 ± 219.8	7390.6 ± 837.4	8894.6 ± 1702.5	17906.7 ± 2759.6
Level 4	2515.1 ± 204.9	9702.4 ± 944.0	14419.1 ± 2637.4	26636.6 ± 3786.3
Level 5	5358.9 ± 385.3	14444.2 ± 1220.8	25602.2 ± 4164.1	45405.3 ± 5770.2
Level 6	18985.1 ± 967.2	34130.1 ± 2914.3	48807.1 ± 9328.2	101922.3 ± 13209.7
Level 7	68482.5 ± 2636.8	69880.6 ± 5836.7	61582.8 ± 11294.3	199945.9 ± 19767.8

Table 7. Zoom levels and predictions

We observe that the estimated counts monotonically increase with the zoom level, which is expected, since the higher the zoom level (higher resolution) is the larger the number of pixels is. Furthermore, since output pixel values were trained to be non-negative (since the prediction output is a density map), it is highly likely that as zoom level increases, the counts will lead to a higher positive bias if we do not use any masking to filter areas that are not regions of interest (ROI). Consider Figure 8 for example. The majority of the images patches in this viewing direction may not include any person at all. ROI does not have to be a precise topology over crowd regions but rather rectangular partition that include at least a few people (not just body parts) would suffice. Clearly, the total estimated count for zoom level 7 is an overestimation — for example, one can easily point out that predicting more than 68,000 people in Figure 8 seems too high even at a first glance. Also, the estimated crowd counts for zoom level 3 seem too low. Then, does the right zoom level exist somewhere between 3 and 7? The answer is no. Table 7 shows that there is no single zoom level which works best for all viewing directions.

Therefore, we adaptively choose a zoom level given a region of an image. To be consistent with the training data (UCF-QNRF), we require each image patch size to be large enough (i.e. zoomed out enough) to have at least 20 people for a given region. Of course, we do not have the ground truth counts. Hence we use predicted counts to adjust the zoom level. If the threshold is not satisfied according to the predicted count, we zoom out, i.e. zoom level decreases and the image patch containing the region of interest grows. If the threshold is satisfied, then we use the lowest predictive variance to decide which zoom level for the region is optimal. By iteratively following the procedure, we form partitions over an image and report predicted counts (predicted density map) per each partition. The sample results are shown in Figures 6, 7, and 8. Each colored square in an image collage represents a sample partition chosen for particular regions. Their corresponding predictive density maps and uncertainty estimates are shown in each subplot.

Left view Est. count	Front view Est. count	Right view Est. count	Total
1202.7 ± 91.8	12408.9 ± 1178.3	34714.5 ± 6207.0	48326.1 ± 7477.1

Table 8. Predictions on CNN giga-pixel images

We acknowledge the prediction on first sample partition (1) in Figure 6 is very challenging and perhaps the predicted count is an underestimation. However, we used zoom level 7 (the highest resolution) for this partition and can't improve further. However, most of other partitions demonstrate plausible density outputs and count estimates. Table 8 report the aggregated count results over the partitions for each viewing direction with the corresponding confidence intervals. Again, in this analysis, we only report on what our method predicted based on what it sees and do not make any inference on what is not seen on the images.

[3]In order to compute an accurate estimate of confidence intervals, perhaps this is the minimal portion that a practitioner can provide labeled data if available. However, for the sake of completeness of our presentation without any ground truth labels provided, we proceed this way.

B. Discussions on Bayesian neural network

In this section we discuss the Bayesian neural network methods, and the justification for the use of bootstrap ensemble to approximate the posterior in this work. Let $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$ be a collection of realizations of i.i.d random variables, where $\boldsymbol{x}_i$ is an image, $\boldsymbol{y}_i$ is a corresponding density map, and N denotes the sample size. In Bayesian neural network framework, rather than thinking of the weights of the network as fixed parameters to be optimized over, it treats them as random variables, and so we place a prior distribution $p(\theta)$ over the weights of the network $\theta \in \theta$. This results in the posterior distribution

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} = \frac{\left(\prod_{i=1}^N p(y_i|x_i,\theta)\right)p(\theta)}{p(\mathcal{D})}.$$

While this formalization is simple, the learning is often challenging because calculating the posterior $p(\theta|\mathcal{D})$ requires an integration with respect to the entire parameter space Θ for which a closed form often does not exist. [38] proposed a Laplace approximation of the posterior. [41] introduced the Hamiltonian Monte Carlo, a Markov Chain Monte Carlo (MCMC) sampling approach using Hamiltonian dynamics, to learn Bayesian neural networks. This yields a principled set of posterior samples without direct calculation of the posterior but it is computationally prohibitive. Another Bayesian method is variational inference [4, 15, 36, 37] which approximates the posterior distribution by a tractable variational distribution $q_\eta(\theta)$ indexed by a variational parameter η. The optimal variational distribution is the closest distribution to the posterior among the pre-determined family $Q = \{q_\eta(\theta)\}$. The closeness is often measured by the Kullback-Leibler (KL) divergence between $q_\eta(\theta)$ and $p(\theta|\mathcal{D})$. While these Bayesian neural networks are the state of art at estimating predictive uncertainty, these require significant modifications to the training procedure and are computationally expensive compared to standard (non-Bayesian) neural networks

[14] proposed using Monte Carlo dropout to estimate predictive uncertainty by using dropout at test time. There has been work on approximate Bayesian interpretation of dropout [14, 24, 39]. Specifically, [14] showed that Monte Carlo dropout is equivalent to a variational approximation in a Bayesian neural network. With this justification, they proposed a method to estimate predictive uncertainty through variational distribution. Monte Carlo dropout is relatively simple to implement leading to its popularity in practice. Interestingly, dropout may also be interpreted as ensemble model combination [58] where the predictions are averaged over an ensemble of neural networks. The ensemble interpretation seems more plausible particularly in the scenario where the dropout rates are not tuned based on the training data, since any sensible approximation to the true Bayesian posterior distribution has to depend on the training data. This interpretation motivates the investigation of ensembles as an alternative solution for estimating predictive uncertainty. Despite the simplicity of dropout implementation, we were not able to produce satisfying confidence interval for our crowd counting problem. Hence we consider a simple non-parametric bootstrap of functions which we discuss in Section 3.1.

C. Ground truth generation

We generate the ground truth density maps by blurring the head annotations provided by the data. This blurring is done by applying a Gaussian kernel (which normalize to 1) to each of the heads in a given image. We use geometry-adaptive kernels [66] to vary the spread parameters of Gaussian depending on local crowd density. The geometry-adaptive kernel is given by:

$$F(z) = \sum_{j=1}^{J} \delta(z - z_j) \times G_{\sigma_j}(z), \text{ with } \sigma_j = \beta \bar{d}_j$$

For each targeted object z_j in the ground truth δ, we use $\bar{d}_j$ to indicate the average distance of k nearest neighbors. To generate the density map, we convolve $\delta(z - z_j)$ with a Gaussian kernel with parameter σ_j (standard deviation), where z is the position of pixel in the image.

D. Comparison on variability

Note that we do not have the ground truth uncertainty to evaluate the predictive uncertainty other than testing whether predictive uncertainty satisfies the definition of confidence interval discussed in Section 3.3. While we recalibrate the estimated uncertainty, we perform a sanity check on the amount of variability of our proposed method before recalibration is applied. We compare our proposed framework with a full bootstrap ensemble, i.e. an ensemble of K independent neural networks. Although hypothetically K bootstrap ensembles can lead to K identical models in the worst cases, due to the nature of highly

non-linear objective in neural network parameter optimization along with random initialization, we should not be worried about this degenerate case. Note that the architectural setting of DUB-CSRNet has a minimal bootstrapping with each output head only branching out at the end of the network architecture, which achieves computational gains but could potentially limit this variability. Hence, we compare our method with a full bootstrap ensemble model which contains the K full-size independent neural networks with each neural network being trained independently. We compute the average of estimated predictive variance on ShanghaiTech Part A and Part B test datasets. In Table 9, we report the predictive variance which is the sum of epistemic and aleatoric uncertainties before recalibration is applied. we observe that DUB-CSRNet shows slightly lower variance than the full bootstrap model, which is expected since the amount of the shared portion of the architecture is much higher for DUB-CSRNet. However, surprisingly the difference in variance is not much given the contrasting network sizes between the full bootstrap model and DUB-CSRNet. Most importantly, with recalibration procedure at hand, we can correct (amplify or shrink) the predictive variance to a suitable amount. Hence, the post-calibrated uncertainty for these two models are almost identical given the same validation dataset and the test dataset.

	Variance	
Method	Part A	Part B
Full-bootstrap CSRNet	47.6	1.17
DUB-CSRNet (Ours)	45.8	1.02

Table 9. Comparison on average predictive variance

E. Note on Inference Runtime

We tested the inference runtime of DUB-CSRNet compared to the vanilla CSRNet to see how much computational increase the proposed bootstrap extension causes. The first set of tests were performed on a CPU with 2.3GHz quad-core Intel Core i5 (8GB RAM). Nvidia P100 GPU was used for the second set of the tests. Each test was performed on ShanghaiTech Part A test dataset. The test showed that the average additional cost is very minimal, with 2% increase on CPU and 0.5% increase on GPU (almost negligible). This makes sense since the output heads only cover the last layer. Also, note that for training since we sample one output node per epoch and update the weights accordingly, there is no additional computation cost per epoch. Hence, this supports our claim that the proposed method is a very efficient way of producing uncertainty estimate.