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 data sets. To the best of our knowledge, we have the best system for ShanghaiTech part A and B, UCF CC 50, UCSD, and 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 [57], monitoring crowds in surveillance systems [11], and counting the number of trees in an aerial image of a forest [15]. 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 [42], image recognition [16], face recognition [45] and image segmentation [30]. Inspired by these successes, many CNN based crowd counting methods have been proposed. Along with density estimation techniques [27], 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 uncertainty in 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 an 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 proposed method achieves the state of the art performances on multiple crowd counting benchmark datasets.
2. Related Work

Many crowd counting studies were developed to solve related real world problems [11, 33, 51]. The previous literature can be categorized into three kinds of approaches depending on methodology: detection-based, regression-based and density-based, which will be briefly reviewed below.

Detection-based crowd counting is an approach to directly detect each of the target objects in a given image or video. A typical approach is to utilize off-the-shelf detectors [26, 28, 56] often using moving-windows [11]. 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 [10, 53]. However, for crowded scenarios, objects are highly occluded and many objects are too small to detect. These issues make detection based approaches infeasible in dense crowd scenes.

Regression-based approaches such as [6, 8, 24, 43, 47] are proposed to bypass the occlusion problem that can be an obstacle for detection-based methods. Specifically, a mapping between image features and the head count is recovered, and the system benefits from better feature extraction and count number regression algorithms [47], [1, 6, 8, 46]. Moreover, [5, 7, 43] leverage spatial or depth information and use segmentation methods to filter the background region and regress count numbers only on foreground segments. These type of methods are sensitive to different crowd density levels and heavily depend on a normalization strategy that is universally good.

Density-based crowd counting, originally proposed in [27], 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. [27] proposes a method which learns a linear mapping between the image feature and the density map. [41] 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 [12, 55, 58]. [60] proposes a multi-column network to directly generate the density map from input images. [54] introduces the boosting process which yield a significant improvement both in accuracy and runtime. To address perspective, [39] feeds a pyramid of input patches into their own designed network, [44] improves over [60] and uses a switch layer to classify the crowd into three classes depending on crowd density and to select one of 3 regressor networks for actual counting [59] jointly estimates the density map and count number with FCN and LSTM layers. [50] uses global and local context to generate high quality density map. [29] introduces the dilated convolution to aggregate multi-scale contextual information and utilizes a much deeper architecture from VGG-16 [48], which at present obtains the state of the art performance.

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 overfitting, so making decisions based on point prediction alone may provide incorrect predictions with spuriously high confidence.

3. Uncertainty in Neural Networks
3.1. Bayesian neural network

Let \( \mathcal{D} = \{(x_i, y_i)\}_{i=1}^N \) be a collection of realizations of i.i.d random variables, where \( x_i \) is an image, \( y_i \) 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

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

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 \( \Theta \) for which a closed form often does not exist. [34] proposed a Laplace approximation of the posterior. [37] 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, 14, 31, 32] which approximates the posterior distribution by a tractable variational distribution \( q_\phi(\theta) \) indexed
by a variational parameter $\eta$. 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

[13] 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 [13, 23, 35]. Specifically, [13] 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 [52] 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 intervals for our crowd counting problem. Hence we consider a simple non-parametric bootstrap of functions which we discuss in the following section.

3.2. 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 data set $\mathcal{D}$ and a function $f_\theta$. We can transform the original dataset $\mathcal{D}$ into $K$ different data sets $\{\mathcal{D}_k\}_{k=1}^K$'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 $f_{\theta_k}$, we train the model on the data set $\mathcal{D}_k$. So each of these models is trained on data from the same distribution but on a different data set. 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 $f_{\theta_k}$.

In cases of using neural networks as base models $f_{\theta_k}$, 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 increases 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.

3.3. Measures of uncertainty

When we address uncertainty in predictive modeling, there are two major sources of uncertainty [21]:

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 [21], 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.
4. Proposed Method

4.1. Single network with K output heads

Training and maintaining a multiple 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 [40]. 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 3.2. 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 [52].

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: \( \theta_0, \theta_1, ..., \theta_K \) \( \sim \mathcal{N}(0, I) \), where \( \theta_i \) 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: \( p(y|f_{\theta}(x)) = \mathcal{N}(f_{\theta}(x), \sigma^2) \), with an observation noise variance \( \sigma^2 \).

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

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

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 \( \sigma^2 \) 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 \( \theta \)), 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:

\[
\text{Var}(y) \approx \sigma^2 + \frac{1}{K} \sum_{k=1}^{K} f_{\hat{\theta}_k}(x) \bar{f}_{\hat{\theta}_k}(x) - \mathbb{E}(y) \mathbb{E}(y) \tag{2}
\]

with approximated predictive mean: \( \mathbb{E}(y) \approx \sum_{k=1}^{K} f_{\hat{\theta}_k}(x) \). 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 \( \sigma^2 \) is constant for all inputs, heteroscedastic regression assumes that \( \sigma^2 \) can vary with input \( x \) [25, 38]. This change can be useful in cases where parts of the observation space might have higher noise levels than others [21]. 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 [21], the network outputs both the estimated density map \( y \) and the noise variance \( \sigma^2 \). 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 \( \sigma^2 \). Let \( \theta_\sigma \) be the parameter corresponding to the output node of the noise variance \( \sigma^2 \). Now, as before, we overload the term \( \theta_k = [\theta_k, \theta_s, \theta_\sigma] \) since \( \theta_s \) 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 \( \sigma^2 \):

\[
[\hat{y}, \sigma^2] = f_{\hat{\theta}_k}(x). \tag{3}
\]

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

\[
\mathcal{L}(\theta) = \frac{1}{D} \sum_i \frac{1}{2\sigma_i^2} ||y_i - \hat{y}_i||^2 + \frac{1}{2} \log \sigma_i^2. \tag{4}
\]

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 \( \sigma^2 \) 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 \( \sigma^2 \) which should be positive, we predict the log variance \( s_i := \log \sigma_i^2 \) instead of \( \sigma^2 \) for the output [21].
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 [29] (CSRNet) which has shown a state of art performance in crowd counting tasks. CSRNet extends the VGG-16 [48] 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 [29]. 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 $\sigma^2$. 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 [48]. 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 $[\hat{y}_n, \hat{s}_n] = f_{\theta_k}(x_n)$ for $n$-th image 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 [22] with a learning rate of $10^{-3}$ is applied to train the model.

4.6. 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 [60] 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 $\delta$, 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 $\sigma_j$ (standard deviation), where $z$ is the position of pixel in the image.

**Algorithm 1 Decomposed Uncertainty using Bootstrap**

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

1. Initialize parameters $\theta$.
2. for each epoch do
3. for all $n = 1$ to $N$ do
4. Sample a bootstrap head $k \sim \text{Uniform}\{1, ..., K\}$
5. Compute predictions $[\hat{y}_n, \hat{s}_n] = f_{\theta_k}(x_n)$
6. Compute loss:
   $$\mathcal{L}(\theta_k) = \frac{1}{D_n} \sum_{i=1}^{N} \frac{1}{2} \exp(\hat{s}_{n,i}) \| y_{n,i} - \hat{y}_{n,i} \|^2 + \frac{1}{2} \hat{s}_{n,i}$$
7. Update $\theta_k$ using gradient $\frac{d\mathcal{L}(\theta_k)}{d\theta_k}$
8. end for
9. end for

5. Experiments

In this section, we first introduce datasets and experiment details. We give the evaluation results and perform com-
comparisons between the proposed method with recent state-of-the-art methods. For all experiments, we used \( K = 10 \) heads for DUB-CSRNet.

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:

\[
\text{MAE} = \frac{1}{N} \sum_{n=1}^{N} |C_n - \hat{C}_n|
\]

\[
\text{RMSE} = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (\hat{C}_n - C_n)^2}
\]

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 an ablation study on ShanghaiTech data to validate the efficacy of our proposed method. We compared with two variants where we either incorporate aleatoric uncertainty only or epistemic uncertainty only. “Epistemic uncertainty only” model refers to a bootstrap ensemble model with a minimization loss defined as Eq. 1. “Aleatoric uncertainty only” model is a single neural network without bootstrap but using a heteroscedastic observation noise as in Eq. 4. We also include CSRNet as our vanilla model. The results in Table 1 show that our proposed method combining both aleatoric and epistemic uncertainty contributes significantly to performance on the evaluation data.

| Method              | MAE  | RMSE |
|---------------------|------|------|
| CSRNet [29]         | 68.2 | 115.0|
| CSRNet + Aleatoric only | 67.8 | 114.3|
| CSRNet + Epistemic only | 67.1 | 113.8|
| DUB-CSRNet (Ours)   | 66.4 | 111.1|

Table 1. Ablation study on uncertainty components

5.3. Performance comparisons

We evaluate our method on four publicly available crowd counting datasets: ShanghaiTech [60], UCF CC 50 [19], UCSD [5], and UCF-QNRF [20].

ShanghaiTech. The ShanghaiTech dataset [60] 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. Ground truth density maps of both subsets are generated with fixed spread Gaussian kernel. Table 2 presents the evaluation results of our method compared to other recent works. The results show that our proposed method achieves the lowest MAE (the highest count accuracy) both in Part A and in Part B. Our method also has the lowest RMSE in Part B and the second lowest RMSE in Part A. Samples of the test evaluations are shown in Figure 3.

UCF CC 50. The UCF CC 50 dataset [58] 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 [29, 58], We use 5-fold cross-validation to evaluate the performance of the proposed method. Ground truth density maps are generated with fixed spread Gaussian kernel. The evaluation results are shown in Table 3. Our proposed method shows the lowest MAE and the second lowest in RMSE.

UCSD. The UCSD dataset [5] 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 (238×158) makes it challenging to generate density maps especially with the use of pooling operations. So we perform upsampling of the images following [29]. MAE and RMSE are evaluated only in the specified ROI during testing. We use frames 601 through 1400 as training set and

| Method     | Part A  |    | Part B  |    |
|------------|---------|----|---------|----|
| Zhang et al. [58] | 181.3  | 277.7| 32.0    | 49.8|
| Marsden et al. [36] | 126.5  | 173.5| 23.8    | 33.1|
| MCNN [60] | 110.2  | 173.2| 26.4    | 41.3|
| Cascaded-MTL [49] | 101.3  | 152.4| 20.0    | 31.1|
| Switch-CNN [44] | 90.4   | 135.0| 21.6    | 33.4|
| CP-CNN [50] | 73.6   | 106.4| 20.1    | 30.1|
| CSRNet [29] | 68.2   | 115.0| 10.6    | 16.0|
| DUB-CSRNet (Ours) | **66.4** | **111.1** | **9.4** | **15.1** |

Table 2. Estimation errors on ShanghaiTech dataset
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 map and the predictive mean of estimated density maps. 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. [18]   | 419.5| 541.6|
| Zhang et al. [58]    | 467.0| 498.5|
| MCNN [60]            | 377.6| 509.1|
| Onoro et al. [39]    | 333.7| 425.2|
| Walach et al. [54]   | 341.4|      |
| Marsden et al. [36]  | 322.8| 397.9|
| Cascaded-MTL [49]    | 318.1| 439.2|
| Switch-CNN [44]      | 295.8| **320.9** |
| CP-CNN [50]          | 266.1| 397.5|
| CSRNet [29]          | 235.2| 332.7|
| DUB-CSRNet (Ours)    | **1.03** | **1.24** |

Table 3. Estimation errors on UCF CC 50 dataset

Table 4. Estimation errors on UCSD dataset

The rest of the frames as testing set following [5]. We generate ground truth density maps with fixed spread Gaussian kernel. The evaluation results are shown in Table 4 and our proposed method shows the lowest MAE and the lowest in RMSE among the methods compared.

**UCF-QNRF.** The UCF-QNRF dataset was recently introduced by [20]. It is currently the largest crowd 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 wide 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. Our method shows the lowest MAE and RMSE.

### 5.4. Discussion on estimated uncertainty

We have observed that our proposed method, DUB-CSRNet, outperforms other state of art methods on almost all benchmark datasets in terms of count estimation. Now, our method also provides uncertainty estimates. Fig. 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 con-
Table 5. Estimation errors on UCF-QNRF dataset

| Method                  | MAE  | RMSE |
|-------------------------|------|------|
| Idrees et al.(2013) [18]| 315  | 508  |
| MCNN [60]               | 277  | 426  |
| Encoder-Decoder [2]     | 270  | 478  |
| CTML [49]               | 252  | 514  |
| Switch-CNN [44]         | 228  | 445  |
| Resnet101 [16]          | 190  | 277  |
| Densenet201 [17]        | 163  | 226  |
| Idrees et al.(2018) [20]| 132  | 191  |
| DUB-CSRNet (Ours)       | 116  | 178  |

Table 6. Comparison on average predictive variance

| Method                  | Part A | Part B |
|-------------------------|--------|--------|
| Full-bootstrap CSRNet   | 82.2   | 1.70   |
| DUB-CSRNet (Ours)       | 80.9   | 1.61   |

5.5. Validation on variability

We validate whether our proposed framework can provide enough diversity of model outputs. Although hypothetically \( K \) bootstrap ensembles can lead to \( K \) identical models in the worst cases, due 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. However, ensuring a reasonable amount of variability over model output is still very essential to our approximation of distributions over functions. Note that the architectural setting of DUB-CSRNet has a minimal bootstrapping with each head only branching out at the end, which achieves computational gains but could potentially limit this variability. Hence, we compare our method with a full bootstrap ensemble model where each neural network is trained independently on average of estimated predictive variance on ShanghaiTech Part A an Part B datasets. In Table 6, we observe that DUB-CSRNet shows variability very close to the full bootstrap model.

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\(^3\) gigapixel images [9] which contains ultra high resolution (64,000 × 64,000 pixels) crowd images.

![Snapshots of prediction on the gigapixel images of the 2017 U.S. presidential inauguration](image)

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 convolutional 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 almost all CNN based crowd counting methods. The extensive experiments demonstrate that the proposed method, DUB-CSRNet, has the state of art performance on all benchmark datasets considered, outperforming previous methods on count accuracy, and produces reasonable uncertainty estimates.

\(^{3}\)Cable News Network
