ZoomCount: A Zooming Mechanism for Crowd Counting in Static Images
Usman Sajid, Hasan Sajid, Member, IEEE, Hongcheng Wang, Senior Member, IEEE, and Guanghui Wang, Senior Member, IEEE

Abstract—This paper proposes a novel approach for crowd counting in low to high density scenarios in static images. Current approaches cannot handle huge crowd diversity well and thus perform poorly in extreme cases, where the crowd density in different regions of an image is either too low or too high, leading to crowd underestimation or overestimation. The proposed solution is based on the observation that detecting and handling such extreme cases in a specialized way leads to better crowd estimation. Additionally, existing methods find it hard to differentiate between the actual crowd and the cluttered background regions, resulting in further count overestimation. To address these issues, we propose a simple yet effective modular approach, where an input image is first subdivided into fixed-size patches and then fed to a four-way classification module labeling each image patch as low, medium, high-dense or no-crowd. This module also provides a count for each label, which is then analyzed via a specifically devised novel decision module to decide whether the image belongs to any of the two extreme cases (very low or very high density) or a normal case. Images, specified as high- or low-density extreme or a normal case, pass through dedicated zooming or normal patch-making blocks respectively before routing to the regressor in the form of fixed-size patches for crowd estimate. Extensive experimental evaluations demonstrate that the proposed approach outperforms the state-of-the-art methods on four benchmarks under most of the evaluation criteria.

Index Terms—Crowd counting, crowd density, cluttered background, decision module, four-way classification, zooming or normal patch-making blocks.

I. INTRODUCTION

In recent years, convolutional neural networks have attracted a lot of attention and been successfully applied to various computer vision problems, such as object detection [24], [29], [46], face recognition [8], depth estimation [16], [17], image classification [7], [50], image-to-image translation [44], [45], and crowd counting [34]. Crowd counting is an integral part of crowd analysis. It plays an important role in event management of huge gatherings like Hajj, sporting, and musical events or political rallies. Automated crowd count can lead to better and effective management of such events and prevent any unwanted incident [18]. Crowd counting is an active research problem due to different challenges pertaining to large perspective, huge variance in scale and image resolution, severe occlusions and dense crowd-like cluttered background regions. Also, manual crowd counting subjects to very slow and inaccurate results due to the complex issues as mentioned above.

To obtain accurate, fast and automated results, CNN-based approaches have been proposed that achieve superior performance over traditional approaches [9], [10], [43]. CNN-based methods can be broadly classified into three categories; regression-based, detection-based, and density map estimation methods. Regression-based methods [41] directly regress the count from the input image. However, these CNN regressors alone cannot handle huge diversity in the crowd images varying from very low to very high. CNN detection-based methods [13], [22] first detect persons in the image and then sum all detection results to yield the final crowd count estimate. Detection-based methods perform well in low crowd images but could not be generalized well to high-density crowd images as detection fails miserably in such cases due to very few pixels per head or person. Density map estimation methods [22], [39], [40] generate density map values, with one value

...
for each image pixel. The final estimate is then calculated by summing all density map values. These methods do not rely on localizing crowd but rather on estimating crowd density in each region of the crowd image. Density map estimation methods outperform other approaches and recent state-of-the-art methods mostly belong to this category. However, density per pixel estimation remains a huge challenge as indicated in [31] due to large variations in the crowd density across different images. This naturally leads to a question: In which scenarios these methods may fail and why?

One key issue with regression and density map methods is that they only rely on direct count estimate and density map estimation per pixel for the input image respectively, thus, they may get subjected to large crowd count for cluttered background image patches. As shown in Fig. 1, models [22] based on these methods consider this $224 \times 224$ image patch as a crowd patch and make false estimates, making the system unreliable as similar patterns are bound to occur in many practical scenarios.

In addition, we observe that both types of methods perform well for images which contain most crowd patches with neither low nor high crowd density. Problem arises when images have most crowd patches with either high or low-density crowd. Due to the limitation in handling such practical diversity in crowd density, these methods may either highly underestimate or overestimate the crowd count in these two extreme cases, as shown in Fig. 1. To further explain this phenomenon, we analyze ten such cases for both extremes separately from very recent UCF-QNRF dataset [22] on the state-of-the-art density map method [3, 22] and direct regression-based method as shown in Fig. 2. It can be observed that, in both extreme cases, the crowd estimates are either highly overestimated or underestimated due to the limitations as discussed above.

To solve these fundamental problems, we propose a modular approach as shown in Fig. 3. It comprises of a Crowd Density Classifier (CDC), a novel Decision Module (DM), and a Count Regressor Module (CRM). The input image is first sub-divided into fixed-size patches ($224 \times 224$) and fed to the CDC module that contains a deep CNN classifier to perform a four-way classification (low, medium, high-density, and no-crowd) on each patch. The classification module eliminates any crowd-like background patches (no-crowd) from the test image and feeds the information about the number of patches belonging to each of the no-crowd, low, medium and high-density classes to the Decision Module (DM) using an accumulator. DM uses either the machine learning based RFDB or heuristic-based Rule-Set Engine (RSE) module to determine if the image belongs to a case of low, normal, or high density. Based on the DM decision, this image is then divided into fixed-size patches using one of three independent image patch-making modules ($Z_{\text{in}}$, Normal, $Z_{\text{out}}$). The image that belongs to low-density extreme case is divided into patches using the zoom-out ($Z_{\text{out}}$) patch-maker; the image that belongs to high-density extreme case is divided via zoom-in ($Z_{\text{in}}$) patch-making block; and the normal case image is split into patches using normal (Normal) patch-maker. These patches are then routed one
by one to the patch-wise count regressor (*COUNTER*) for
crowd estimate and the image total crowd count is obtained
by summing all patches count.

The $Z_{in}$ block divides each input patch into four $112 \times 112$
patches, and then up-scales each patch by $2 \times$ before routing
each patch to the count regressor. Intuitively, this module is
further zooming-in into the image and looking in-detail all
patches by using $1/2$ input patch size instead of the original
$224 \times 224$ patches. Similarly, zoom-out patch-maker divides
the input image into $448 \times 448$ patches, and down-scales each
patch by $2 \times$ as it is dealing with the image containing low-
density crowd patches mostly. The normal case image directly
employs the original $224 \times 224$ patch size with no up-scaling
or down-scaling. The main contributions of this work include:

- The paper reveals and analyzes the fact that extremely
  high and low dense crowd images greatly influence
  the performance of the state-of-the-art regression and
density map based methods for crowd counting.

- A novel strategy is proposed to address the problem of
counting in highly varying crowd density images by first
classifying the images into either one of the extreme
cases (of very low or very high density) or a normal
case, and then feeding them to specifically designed
patch-makers and crowd regressor for counting.

- A novel rule-set engine is developed to determine
whether the image belongs to an extreme case. For
images of extremely high density, a zoom-in strategy is
developed to look into more details of the image; while
for images of low-density extreme, a zoom-out based
regression is employed to avoid overestimate.

- We created four new datasets, each from the corre-
sponding crowd counting benchmark, for the training
and testing of different machine learning algorithms to
classify an image as normal, high or low dense extreme
case using its patches classification count. These man-
ually verified datasets will facilitate the researchers in
analyzing complex crowd diversity, which is at the core
of the crowd analysis.

The proposed ZoomCount scheme is thoroughly evaluated
on four benchmarks: UCF-QNRF [22], ShanghaiTech [49],
WorldExpo’10 [48], and AHU-Crowd [19]. The experimental
results demonstrate the effectiveness and generality of the
proposed strategy and rule-sets, which are never realized for
crowd counting. The overall performance of the proposed
model outperforms the state-of-the-art approaches on most of
the evaluation criteria. The proposed models and source code,
as well as the created datasets, will be available on the author’s
website.

II. RELATED WORK

Crowd counting is an active research area in computer
vision with different challenges related to large perspective,
occlusion, cluttered background regions and high variance in
crowd density across different images. Earlier work [42], [43]
focused on the head or full-body detection for counting using
handcrafted features for detectors learning. These methods
failed in case of high dense images, where it is hard to
find such handcrafted features. The approaches were shifted
towards regression based counting [9], [10], [33], where a
mapping function was learned to directly regress count from
local patches of an image. These methods improved the
counting process, however, they could not handle huge crowd
diversity and also lack awareness about crowd density across
all parts of the image.

Recently, CNN-based approaches have been widely used
[22], [26], [27], [41]. They are broadly categorized into three
classes; Counting by detection, counting by direct regression,
and counting using density map estimation. CNN-based object
detectors [13], [32] detect each person in the image, and
the final count is then calculated by summing all detections.
These methods [25], [36] deteriorate in high density and severe
occlusion cases, where each head only occupies a few pixels.
Counting by direct regression methods [41] directly regress
count by learning feature maps from the input image patch.
Wang et al. [41] proposed an end-to-end AlexNet [23] based
regressor for crowd count. These methods alone cannot handle
huge diversity in different crowd images.

Density map estimation methods [6], [22], [26], [27], [39]
learn to map crowd density per pixel of an image without
localizing the counts. The final estimate is calculated by
summing all density estimations. Zhang et al. [49] proposed
a three-column CNN architecture (MCNN) to handle crowd
diversity across images. Each column is designed to handle
different scales using different receptive field sizes. Sindagi
et al. [40] extended the idea of MCNN to incorporate con-
textual information for high-quality density maps generation.
Recently, Sam. et al. [35] proposed SwitchCNN which routes
each input patch to one of three independent CNN regressors
using a switch CNN classifier. Based on the classification and
regression idea, Sindagi et al. [39] designed a Cascaded-MLT
that estimates count for the whole image by using cascaded
10-way classification prior and final density map estimation.

Crowd counting models, based on whole image estimation
and training from the scratch, are subjected to over-fitting due
to limited dataset availability (only a few hundred training
images). Thus, patch-based models are widely used nowadays.
The final sum is computed by adding up all patch count
estimates. Liu et al. [27] proposed a hybrid approach by
incorporating both regression and detection blocks using an
attention-guided mechanism to handle low and high-density
cases simultaneously. Li et al. [29] designed a CSRNet to get
multi-scale contextual information by incorporating dilation-
based convolutional layers. Idrees et al. [22] proposed a com-
position loss based model for simultaneous crowd counting
and localization. Existing methods perform worse in extreme
cases where most crowd patches belong to either high density
or low density. Moreover, these methods lack the ability to
fully discard any cluttered background regions in the image,
thus resulting in overestimate.

III. PROPOSED APPROACH

The proposed framework is shown in Fig. 3 which is
composed of three modules namely Crowd Density Classifier
(CDC), Decision Module (DM) and Count Regressor Module (CRM). The input image is first subdivided into $224 \times 224$ size patches and each patch then passes through the CDC module for 4-way classification (low, medium, high-density or no-crowd). The accumulator gathers and feeds patch count per class information to the Decision Module. Based on accumulator information and utilizing either Random Forest based Decision Block (RFDB) or heuristic-based Rule-Set Engine (RSE), DM routes this image to one of three specialized patch-making blocks ($Z_{in}, Normal, Z_{out}$) of CRM where the input image is divided into corresponding patches, followed by the crowd estimate for each patch via crowd regressor ($COUNTER$). Finally, the image crowd count is calculated by summing all patches count. Below we will discuss the details of each module, as well as the rules defined for the two possible extremes.

A. Crowd Density Classifier (CDC) Module

The CDC module is composed of a deep CNN 4-way classifier that specializes in making a distinction between no-crowd (NC), low-density (LC), medium-density crowd (MC), and high-density crowd (HC) for each input patch. Let $X$ be a test image sub-divided into $N$ patches $[x_1, x_2, ..., x_N]$, each with a size of $224 \times 224$. The accumulator gathers each patch classification result for the input image $X$ as follows:

$$P_y + = 1, \text{ if class}(x_i) = y \quad (1)$$

for $i = 1, 2, ..., N$ and $y$ belongs to either NC, LC, MC or HC class label. In the end, the accumulator passes the patch count per class ($PCC_X$) of this image to the decision module (DM) as:

$$PCC_X = \{P_{NC}, P_{LC}, P_{MC}, P_{HC}\} \quad (2)$$

where $P_{NC}, P_{LC}, P_{MC}$ and $P_{HC}$ denote the total number of patches being classified as NC, LC, MC and HC respectively of the image $X$. Patches being classified as NC are discarded, and thus remaining $\{N - P_{NC}\}$ crowd patches are going to be used for final crowd estimate. As a result, the crowd-like cluttered background regions (such as the tree leaves shown in Fig. 1), which may result in overestimation otherwise, will be eliminated.

Definitions of NC, LC, MC and HC class labels. During experiments for each crowd counting benchmark dataset, we randomly extract patches from its training images for the CDC classifier training and assign a ground truth class label (NC, LC, MC, HC) to each extracted patch. Since these datasets also contain the localization of people, so we generate the ground truth class label for each patch using this information and the maximum people count possible in any image patch of the corresponding dataset. LC class label is assigned to a patch if the ground truth people count for that patch is less than or equal to 5% of the maximum possible count but greater than zero as zero crowd means NC class patch. Similarly, patches with ground truth people count between 5% to 20% of the maximum possible count are assigned the MC class label, while patches containing more than 20% of the maximum people count are labeled as HC category patches. In the end, a total of 90,000 patches, with an equal amount per class label, are generated for the CDC classifier training in each benchmark setting. Example patches for each class label are shown in Fig. 4.

Classifier Details. We use DenseNet-201 [20] as our 4-way classifier, as shown in Fig. 5. It has four dense blocks with transition layers (convolution and pooling) in between them to adjust feature maps size accordingly. The DenseNet-201 has consecutive $1 \times 1$ and $3 \times 3$ convolutional layers in each dense block in $\{6, 12, 48, 32\}$ sets respectively. At the end of the last dense block, a classification layer is composed of $7 \times 7$ global average pooling, followed by $1000 - D$ fully connected layer and the final 4-way softmax classification with cross-entropy loss.

B. Decision Module (DM)

The decision module, based on the CDC module output $PCC_X$, decides if the test image should be treated as a normal image or a low or a high-density extreme case image. DM makes this decision by utilizing one of the two separate and independent decision-making blocks, namely Rule-Set Engine (RSE) and Random Forest based Decision Block (RFDB). RSE is a novel heuristic-based approach which employs the rule-sets to detect if the test image is either an extreme or a normal case, while RFDB is an automated decision-making block based on Random Forest algorithm that learns to map the test image features ($P_{NC}, P_{LC}, P_{MC}, P_{HC}$) to the respective class label ($Z_{in}, Normal, Z_{out}$). We also create new RFDB training datasets, each from corresponding crowd counting benchmark, for the training of RFDB module as explained in Sec. III-B3.

1) Rule-Set Engine (RSE): The accumulated patch count per class ($PCC_X$) from CDC module is tested against two different rule-sets to determine if an input image is a case of low or a high density extreme or a normal one so that
TABLE I
DESCRIPTION OF TWO RULE-SETS: THE LOWER DENSITY EXTREME (RULES 1-4) AND THE HIGHER DENSITY EXTREME (RULES 5-8). THIRD COLUMN INDICATES IMAGES THAT ARE AFFECTED THE MOST (IN TERMS OF RESOLUTION) BY THAT RULE. SOME RULES HAVE MUCH HIGHER TENDENCY TO BE APPLIED ON THE LOWER RESOLUTION (LR) OR HIGHER RESOLUTION (HR) IMAGES, WHEREAS SOME RULES HAVE IMPACT ON ALL TYPES OF IMAGES (INDICATED BY ‘MIX’).

| Extreme Case Type | Rule | Most Affected Images | Description |
|-------------------|------|----------------------|-------------|
| Low density       | 1    | LR                   | Image contains LC and NC patches only. |
|                   | 2    | Mix                  | Image should have LC patches and no HC patch. |
|                   | 3    | HR                   | Image has more than 50% patches being classified as LC category. |
|                   | 4    | Mix                  | At most 5% patches belong to HC category with at least one patch from NC category. |
| High density      | 5    | Mix                  | Image with all patches belonging to HC category only. |
|                   | 6    | Mix                  | All patches are MC category only. |
|                   | 7    | LR                   | More than 50% patches of the image are from HC category. |
|                   | 8    | Mix                  | Image should have NC patches and at least 33% or more from both $P_{HC}$ and $P_{MC}$ category each. Intuitively, first condition of R8 emphasizes the fact that more no-crowd patches shift image towards high dense case, if supported by other given conditions. |

Algorithm 1 Rule-Set Engine Algorithm

Input: $PCC_{X}(\text{Patch Count per Class for Test Image } X) = \{P_{NC}, P_{LC}, P_{MC}, P_{HC}\}$

Output: Normal or $Z_{in}$ or $Z_{out}$

if input patch count satisfies any of following rules then

Output = $Z_{out}$

Rule 1: if $P_{HC} + P_{MC} = 0$
Rule 2: if $P_{HC} = 0$ and $P_{LC} > 0$
Rule 3: if $P_{LC} > (P_{all} \ast 0.50)$
Rule 4: if $P_{NC} > 0$ and $P_{HC} <= (P_{all} \ast 0.05)$

end

else if input patch count satisfies any of following rules then

Output = $Z_{in}$

Rule 5: if $P_{LC} + P_{MC} = 0$
Rule 6: if $P_{LC} + P_{HC} = 0$
Rule 7: if $P_{HC} > (P_{all} \ast 0.50)$
Rule 8: if $P_{NC} > 0$ and $P_{MC} >= (P_{all} \ast 0.33)$ and $P_{HC} >= (P_{all} \ast 0.33)$

end

else Output = Normal

It can be divided into patches using the most suitable patch-making block ($Z_{in}, Normal, Z_{out}$). The overall goal of RSE is to encourage an image with more number of high-density patches to pass through zoom-in patch-making block ($Z_{in}$), whereas the image with more number of low-density patches goes through a zoom-out patch-making block ($Z_{out}$). If the image does not belong to any of the two extreme cases, it will be treated as a normal case that uses the normal patch-maker ($Normal$).

Rules. The RSE module consists of two generalized rule-sets, aiming to detect the images belonging to any of the two extreme cases: the low-density extreme (Rules 1-4) and the high-density extreme (Rules 5-8). As illustrated in Algorithm 1 if no rule applies to the test image $X$, it will use $Normal$ patch-maker, whereas the image satisfying any rule from (1 - 4) or (5 - 8) will generate its patches using $Z_{out}$ or $Z_{in}$ patch-making blocks respectively. Each rule is explained in detail in Table I. This table also shows the most affected images by a specific rule in terms of resolution. For example, Rule 7 is highly applicable on relatively lower resolution (LR) images, whereas Rule 2 can affect images of any resolution equally. It is important to note that these rule-sets are used consistently and evaluated across all four publicly available datasets in the experiments, thus demonstrating the generality and efficacy of such rule-sets. In addition, the current rule sets are extendable by adding more rules to refine the classification/decision process. Please note that all parameters in Table I are chosen empirically.

2) Random Forest based Decision Block (RFDB): The scalable rule-sets based decision process yields promising results as demonstrated throughout the experiments in Sec. V. Nevertheless, there are many heuristics to handle and it requires manual input and special attention while inducting new rules. To address this issue, we propose an automated machine learning based approach that learns the decision process by mapping the four features ($P_{NC}(\%)$, $P_{LC}(\%)$, $P_{MC}(\%)$, $P_{HC}(\%)$) to respective class label ($Z_{in}$, $Normal$ or $Z_{out}$) for each image, where the features denote percentages instead of total image patches belonging to NC, LC, MC and HC classes respectively and labels represent zoom-in, normal and zoom-out patch-making blocks required to generate the patches from the particular input image before proceeding to the count regressor. We employ percentages for features because of the huge variance in resolution across different images in a dataset, which directly influences the features and hence training quality. In addition, since there is no such dataset available for the crowd counting problem to-date that can help in learning this mapping, thus we generate a new RFDB training dataset from each corresponding benchmark as explained in detail in next subsection. To automate the decision block process, we explored different machine learning classification models and found the random forest-based model to be the most effective as demonstrated in the experiments in Sec. V. Thus, we choose the random forest algorithm and hence named this module as Random Forest based Decision Block.

Random Forest (RF), being a bootstrap aggregation or bagging based ensemble method, can be used both for classifi-
cration and regression. We employ the RF algorithm to classify the four features \( (P_{NC}(\%), P_{LC}(\%), P_{MC}(\%), P_{HC}(\%)) \) to a class label of \( (Z_{in}, \text{Normal or } Z_{out}) \) by building, training and tuning a large collection of de-correlated binary decision trees. Each tree then casts a vote for class prediction for the test sample. Finally, the class label with a majority vote is assigned to that test sample i.e., the input image.

Each RF decision tree \( t_k \) is built using a bootstrap sample \( BS(t_k) \) which is generated from the training data. Such bootstrap sample is given as:

\[
BS(t_k) = \begin{bmatrix}
NC_1 & LC_1 & MC_1 & HC_1 & C_1 \\
NC_2 & LC_2 & MC_2 & HC_2 & C_2 \\
NC_3 & LC_3 & MC_3 & HC_3 & C_3 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
NC_M & LC_M & MC_M & HC_M & C_M
\end{bmatrix}
\]  

(3)

for \( K = 0, 1, 2, ...N - 1 \), where \( N \) denotes the total number of RF trees. Each row represents one training sample for the tree \( t_k \) with the class label as the last entry. We use \( N = 100 \), which is set empirically as no significant improvement has been observed in performance beyond this number. The trees are grown using the classification and regression (CART) algorithm, where the nodes get split until all leaves become unmixed or contain less than \( m_{min} \) samples \([1]\). We use \( m_{min} = 2 \) throughout our experiments, thus splitting nodes until they contain either only one sample or become pure. To quantify the quality of a tree node split, \textit{Gini Impurity} has been used as:

\[
\text{Gini Impurity}_n = \sum_{i=1}^{L=3} -F_i(1 - F_i)
\]  

(4)

where \( L \) denotes the total unique class labels and \( F_i \) denotes the frequency of class label \( i \) at node \( n \). During testing, each RF tree gives its class prediction for test image \( X \). Final class label is obtained by the majority vote criterion \([14]\) as follows:

\[
C_{RF}(X) = \text{majority vote}(C_k(X))_{1}^{N}
\]  

(5)

where \( C_k(X) \) represents the class prediction by the \( k^{th} \) RF tree.

Feature Importance (FI) depicts the role of each feature in determining the node split and eventually the quality of the RF decision trees building. Features with much lesser FI value can be easily discarded as they do not play any significant role in decreasing the node impurity. As shown in the left graph in Fig. 6, all four features have approximately the same FI values in each RFDB dataset. Thus, we keep and use all four available features \( (P_{NC}(\%), P_{LC}(\%), P_{MC}(\%), P_{HC}(\%)) \) in all four newly generated RFDB datasets.

3) Dataset generation for RFDB: The RFDB module learns to map the image extracted features \( (P_{NC}(\%), P_{LC}(\%), P_{MC}(\%), P_{HC}(\%)) \) to the respective class label \( (Z_{in}, \text{Normal or } Z_{out}) \) using training dataset with the corresponding mapping. No such dataset has been created to-date. Thus, for each benchmark (ShanghaiTech \([49]\), UCF-QNRF \([22]\), AHU \([19]\), we created a new respective RFDB dataset which contains this mapping.

To create the new RFDB dataset, each training image’s required features \( (P_{NC}(\%), P_{LC}(\%), P_{MC}(\%), P_{HC}(\%)) \) are extracted using ground truth crowd localization information and definitions of class labels \( (NC, LC, MC, HC) \) as stated in \([11-A]\) followed by manual verification and ground truth (GT) class label assignment. To ensure the quality of the generated dataset, each sample entry was then double checked for any inconsistency, duplicates, missing and erroneous cases. For the extracted features, we use percentages instead of the actual number of patches \( (P_{NC}, P_{LC}, P_{MC}, P_{HC}) \) belonging to each category because of the huge resolution difference across the images within each benchmark dataset.

Statistics. For each of the four crowd counting benchmarks, we create the corresponding RFDB dataset using its corresponding training images. For instance, in the case of ShanghaiTech dataset (300 training images), we generate the new 300 samples RFDB dataset with each entry being created using one of the respective training image, followed by manual verification that also includes removal or modification of inconsistent entries. In total, 220, 2830 and 812 samples are finalized for the three RFDB datasets based on ShanghaiTech \([49]\), WorldExpo’10 \([48]\) and UCF-QNRF \([22]\) benchmarks, respectively. For AHU \([19]\) based RFDB dataset, 90 out of 96 available entries are kept on average with 5-fold cross-validation. The lower graph in Fig. 6 shows the percentage of each class label in all four newly created RFDB datasets.
C. Count Regressor Module (CRM)

The CRM module comprises three independent patch-making blocks and a deep CNN count regressor (COUNTER). The decision module routes the test image to one of these patch-makers for dividing it into $224 \times 224$ patches after required up-scaling or down-scaling, followed by the crowd count for each image patch via the count regressor (COUNTER). The regressor employs DenseNet-201 inspired architecture with a single neuron after the fully connected layer to directly regress the crowd count. Mean squared error (MSE), as defined below, has been employed as the loss function for the count regressor $c$:

$$ L_c = \frac{1}{N} \sum_{i=1}^{N} (F(X_i, \Theta) - Y_i)^2 $$

where $N$ is the number of training patches per batch, $Y_i$ is the ground truth crowd count for the input patch $X_i$, and $F$ is the function that maps the input patch $X_i$ to the crowd count with learnable parameters $\Theta$.

Zoom-in based Patch Maker ($Z_{in}$): Ideally, the decision module (DM) routes the image, with most crowd patches being classified as high-density crowd, to this patch-maker. The image, using this patch-maker, is further sub-divided into equal $112 \times 112$ patches, and then up-scaled by $2 \times$ before proceeding to the count regressor for each patch crowd count. Intuitively, it looks into each patch in detail by estimating the count on smaller zoomed-in highly crowded patches. In this way, it greatly stabilizes and improves the count estimate for high-density images, where other methods may either underestimate or overestimate too much due to fixed patch sizes, as demonstrated in the experiments Sec. V.

Zoom-out based Patch Maker ($Z_{out}$): This block is responsible for handling the low-density extreme case images as detected and routed by the decision module. $Z_{out}$ takes $448 \times 448$ original patches of the test image $X$, down-scales them by 2 times, and feeds each resultant patch to the CDC classifier to eliminate any no-crowd patches, as shown in Fig. 7. The count estimate for each crowd patch is then computed through CRM count regressor (COUNTER) followed by the image total count estimate, which is the sum of all patches crowd counts. In other words, it assists the count regressor by using larger area per input patch ($448 \times 448$ down-scaled to $224 \times 224$) which alleviates the overestimation problem.

Normal case: In Normal case, the images are divided into $224 \times 224$ size patches with no up- or down-scaling before patch-wise count regression. It is also worth mentioning that there is no need to explicitly look for and eliminate any no-crowd patches in case of Normal and $Z_{in}$ case images as such background patches are automatically removed during the CDC module classification process, and thus we can also reuse the remaining CDC module crowd patches in both cases for crowd estimate.

The input image that is classified as Normal case, may contain mixed crowd numbers in different regions. Empirically, it has been observed that the deep CNN based crowd counter (i.e. the CRM regressor) can directly handle such images effectively without any rescaling or classification process.

Thus, this work only focuses on the images of extreme case (low-density or high-density) that contain most of the regions with same crowd level (low- or high-density), since these extremes have huge influence on crowd count, and a special attention to these cases will significantly mitigate the over- or under-estimation issue as discussed in the introduction.

D. ZoomCount and Switch-CNN Comparison

Switch-CNN \cite{55} is one of the state-of-the-art crowd counting models, and it classifies the input image each patch into one of three crowd-density labels (low, medium, and high). However, our approach employs this classification process fundamentally different from Switch-CNN. The key differences and comparison are as follows:

- **Crowd-Density Classification Purpose and Usage.** Switch-CNN primarily uses the crowd-density classification to route the input patch to one of three specialized count regressors. Subsequently, crowd estimation for each patch is being done using one of these deep count regressors. On the other hand, we employ the crowd-density classification process to facilitate the Decision Module (DM) in detecting whether the input image is either one of the two extremes (low- or high-density) or a normal case image. Then, the patches are rescaled accordingly before proceeding to the only count regressor for the crowd estimation. Thus, the underlying usage and purpose of crowd classification completely differ in both methods.

- **Patch-based vs Image-based Decision.** Based on the crowd-density classification as discussed in the above point, Switch-CNN selects the appropriate crowd count regressor individually for each input patch (patch-wise decision). While ZoomCount selects the most appropriate rescaling operation suitable for the whole image (image-based decision) using this information. Thus, our method also takes the global context of the image into consideration during the decision-making process instead of relying solely on the local image patches.

In addition, Switch-CNN trains using multiple complex training steps including pretraining, differential training, switch training, and coupled training, while ZoomCount selects the most appropriate rescaling operation suitable for the whole image (image-based decision) using this information. Thus, our method also takes the global context of the image into consideration during the decision-making process instead of relying solely on the local image patches. Moreover, our method outperforms the Switch-CNN under all evaluation metrics as demonstrated in the experiments section.
TABLE II
Benchmark datasets (used in the experiments) statistics.

| Dataset            | Images | Annotations | Min | Max. | Avg. |
|--------------------|--------|-------------|-----|------|------|
| UCF-QNRF [22]      | 1355   | 1,251,642   | 65  | 12865| 815  |
| ShanghaiTech Part-A [49] | 482    | 241,677     | 33  | 3139 | 501  |
| WorldExpo’10 [48] | 3980   | 225,216     | 1   | 334  | 56   |
| AHU-Crowd [19]    | 107    | 45,807      | 58  | 2201 | 428  |

TABLE III
Comparison of ZoomCount with the state-of-the-art methods on UCF-QNRF [22] dataset. Methods with “*” do not use density maps at all. Both versions of our method outperform the state-of-the-art on most of the evaluation criteria.

| Method                | MAE  | MNAE | RMSE |
|-----------------------|------|------|------|
| Idrees et al. [21]*   | 315  | 0.63 | 508  |
| MCNN [49]             | 277  | 0.55 | 426  |
| Encoder-Decoder [5]   | 270  | 0.56 | 478  |
| CMTL [39]             | 252  | 0.54 | 514  |
| SwitchCNN [35]        | 228  | 0.44 | 445  |
| Resnet101 [15]*       | 190  | 0.30 | 277  |
| Densenet201 [20]*     | 163  | 0.40 | 226  |
| CL [22]               | 132  | 0.26 | 191  |
| ZoomCount-RSE*        | 130  | 0.23 | 204  |
| ZoomCount-RFDB*       | 128  | 0.20 | 201  |

IV. IMPLEMENTATION DETAILS

A. Training Details

The CDC classifier and the count regressor (COUNTER) expect fixed size patch of $224 \times 224$ as the input. For both modules, we randomly extract $112 \times 112$, $224 \times 224$ and $448 \times 448$ patches from the training images. Around 90,000 such patches with mixed crowd numbers are generated for each of these modules. The count regressor is trained for 80 epochs with Adam optimizer and a batch size of 16 and starting learning rate of 0.001, decreased by half after every 20 epochs. The classifier employs the stochastic gradient descent (SGD) based optimization with multi-step learning rate starting at 0.1 and decreased by half after 25% and 50% epochs with 80 epochs in total. For each dataset, around 10% training data has been used for validation as recommended in the corresponding literature. For the random forest algorithm in RFDB, we utilize machine learning library scikit-learn for python programming. The Random Forest model was trained using 100 RF decision trees, where each RF tree is trained using the bootstrapped sample with Gini Impurity as node split quality criterion. 10% of the training data has been used for validation in case of each RFDB dataset.

B. Evaluation Details

In order to make a fair and consistent comparison with other methods, we employ three evaluation metrics namely Mean Absolute Error (MAE), Mean Normalized Absolute Error (MNAE) and Root Mean Squared Error (RMSE) defined as below:

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} |Y_i - \hat{Y_i}| \quad (7)$$

$$\text{MNAE} = \frac{1}{N} \sum_{i=1}^{N} \frac{|Y_i - \hat{Y_i}|}{Y_i} \quad (8)$$

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (Y_i - \hat{Y_i})^2} \quad (9)$$

where $N$ denotes the total number of test images, and $Y_i$ and $\hat{Y_i}$ are the ground truth and the estimated counts respectively for the test image $i$.

V. EXPERIMENTS

In this section, we demonstrate both quantitative and qualitative results from extensive four benchmark datasets: UCF-QNRF [22], ShanghaiTech [49], WorldExpo’10 [48], and AHU-Crowd [19]. These datasets contain images with huge crowd variance, different camera perspective, and complex
cluttered background regions. Details about each benchmark are given in Table [II]. At the end of this section, we also discuss computational time analysis and compare that with state-of-the-art methods.

Two different versions of the proposed model, the Rule-Set Engine (ZoomCount-RSE) based and the automated RFDB module (ZoomCount-RFDB) based version, are being compared separately with the state-of-the-art techniques throughout this section. Both ZoomCount versions give almost identical and much better performance under most of the evaluation criteria on the four benchmark datasets.

A. Experiments on UCF-QNRF Dataset

The dataset was recently published by Idrees et al. [22], which is a challenging and the first dataset of its kind. On one hand, it contains images with resolution as high as (6666 × 9999) and as low as (300 × 377); on the other hand, crowd count per image range from a maximum value of 12,865 to a minimum count of 65. The total number of annotations in this dataset is 1,251,642, indicating the level of crowd complexity. It contains 1535 images in total, out of which 1201 and 334 images are used for training and testing respectively. We compare ZoomCount with the state-of-the-art methods and tabulate the results in Table [III]. It is evident that both versions of our method outperform all other approaches in terms of MAE and MNAE; while performing comparatively closer to the best in terms of RMSE.

In order to evaluate the influence of different rules, we perform the ablation experiments, as shown in Table [IV]. We analyze the effect of all rules (R1 to R8) by removing them one at a time in ZoomCount-RSE version of the proposed method. As shown in the results, doing so greatly decreases the performance of our method, thus demonstrating the importance of those rules. We also analyze the effect of removing both of the zoom-in and zoom-out patch-makers in the experiments. From the results in Table [IV], it is evident that both modules play an effective role in improving the overall performance of our method. The last three columns of Table [IV] show the number of the test images passed through the zoom-in, normal and zoom-out patch-makers respectively. In the original setting, 75 (~ 22%) images passed through the zoom-in patch-maker, whereas the zoom-out block handled 162 (~ 48.5%) images and normal patch-maker was used only for 97 (~ 29.5%) images, showcasing quantitative importance of these extreme case handlers, as shown in Fig. 8. We also compare the crowd estimate of ten test images each, for both extreme cases with DenseNet [20] direct regression and the state-of-the-art CL [22] density map method. Our method performs much better in both cases, as shown in Fig. 2.

B. Experiments on ShanghaiTech Dataset

The ShanghaiTech part A dataset contains a total of 482 images with 241,677 annotations, randomly collected from the internet, with a split of 300 and 182 images for training and testing respectively. We compare our method with the state-of-the-art methods as shown in Table [V]. The results show that our method outperforms all other methods on every evaluation metric with significant improvement from 0.224 to 0.190 (~ 15%) in terms of MAE and from 104.5 to 94.5 (~ 9.6%) in case of RMSE.

The proposed rules (R1-R8) play an important and effective role in the performance improvement of ZoomCount-RSE version of our method as shown in Table [VI] where we remove each rule one at a time. It is clear that the error increases by removing these rules. In the same table, We also analyze the effect of removing the zoom-in and zoom-out blocks separately and together. As expected, the performance plunges dramatically as error increases without using them. The last three columns show the number of test images passing through the zoom-in, normal and zoom-out patch-makers respectively. In the original setting, 21 (~ 11.5%), 121 (~ 66.5%) and 40 (~ 22%) images are handled by the zoom-in, normal and

### Table V

| Method             | MAE | MNAE | RMSE |
|--------------------|-----|------|------|
| Zhang et al. [48]  | 181.8 | - | 277.7 |
| MCNN [49]          | 110.2 | - | 173.2 |
| Cascaded-MTL [39]  | 101.3 | 0.279 | 152.4 |
| Switch-CNN [35]    | 90.4 | - | 135.0 |
| CP-CNN [40]        | 73.6 | - | 106.4 |
| CSRNet [26]        | 68.2 | - | 115.0 |
| IG-CNN [17]        | 72.5 | - | 118.2 |
| L2R [25]           | 72.0 | - | 106.6 |
| ICC [31]           | 68.5 | - | 116.2 |
| SA-Net [6]         | 67.0 | - | 104.5 |
| Deep-NCL [47]      | 73.5 | - | 112.3 |
| Densenet201 [20]   | 79.3 | 0.224 | 118.9 |
| ZoomCount-RSE*     | 66.6 | 0.197 | 94.5 |
| ZoomCount-RFDB*    | 66.0 | 0.190 | 97.5 |

### Table VI

| Without | MAE  | MNAE  | RMSE  | ZoomCount-RSE* | ZoomCount-RFDB* |
|---------|------|-------|-------|----------------|-----------------|
| R1      | 66.8 | 0.199 | 95.3  | 21 | 135 | 26 |
| R2      | 66.8 | 0.198 | 96.0  | 21 | 122 | 39 |
| R3      | 67.2 | 0.198 | 94.7  | 21 | 124 | 37 |
| R4      | 68.0 | 0.206 | 95.9  | 21 | 143 | 18 |
| R5      | 69.4 | 0.200 | 103.7 | 15 | 127 | 40 |
| R6      | 69.1 | 0.210 | 101.8 | 19 | 123 | 40 |
| R7      | 66.8 | 0.200 | 97.2  | 20 | 122 | 40 |
| R8      | 69.4 | 0.199 | 97.2  | 09 | 133 | 40 |
| ZoomIn  | 74.9 | 0.200 | 116.4 | 0 | 142 | 40 |
| ZoomOut | 69.1 | 0.210 | 96.5  | 21 | 161 | 0 |
| ZoomIn & ZoomOut | 79.3 | 0.224 | 118.9 | 0 | 182 | 0 |
| -       | 66.6 | 0.197 | 94.5  | 21 | 121 | 40 |
zoom-out blocks respectively, thus proving the quantitative importance of all of them and associated rules in ZoomCount-RSE, as shown in Fig. 8. In Fig. 9 we analyze the performance of our method on the average count across image groups with different total crowd counts. As compared with the state-of-the-art methods, ZoomCount performs the best in most cases.

C. Experiments on WorldExpo’10 Dataset

The WorldExpo’10 [45] is a large dataset, composed of 1132 video sequences taken by 108 different cameras. The training set consists of 3380 images from 103 different scenes, whereas the testing set has 5 scenes with a total of 600 frames. This benchmark also consists of Region of Interest (RoI) and perspective maps. We only utilize the RoIs in the images during training and testing stages. The MAE based evaluation results on the five test scenes and the average MAE error are shown in Table VII. As shown, the proposed model achieves the lowest average MAE and the best performance on two test scenes (S1 and S3) and the average MAE metric, while perform reasonably closest to the best for other test scenes.

D. Experiments on AHU-Crowd Dataset

AHU-Crowd [19] dataset contains 107 images with 45,807 human annotations. The crowd count ranges from 58 to 2201 per image. As per the standard being followed for this dataset [19], we performed 5-fold cross-validation and evaluated our method using the same three evaluation metrics. ZoomCount outperforms all the other methods as shown in Table VIII. It is worth-mentioning that ZoomCount decreases MAE and MNAE significantly by \( \sim 28\% \) (81 to 74.9) and \( \sim 45\% \) (0.199 to 0.140) respectively, whereas RMSE decreases drastically by \( \sim 11.2\% \) (124.9 to 111).

E. CDC Classifier and CRM Regressor Architecture Selection

Choosing an appropriate network architecture for the CDC classifier and the CRM regressor is essential for the effectiveness of the proposed scheme. In this section, we analyze different state-of-the-art architectures for this objective, including VGG-16 [38], ResNet-101 [15], and DenseNet-201 [20]. For the classifier choice evaluation, the final 1000-way classification layer in VGG-16 and ResNet-101 is replaced with a 4-way classification layer. Similarly, for the CRM regressor evaluation, the final FC layer in VGG-16 and ResNet-101 networks is followed by a single neuron to directly regress
TABLE X
THE CLASSIFICATION ACCURACY RESULTS OF THE RSE AND THE RFDB BASED VERSIONS OF THE DM MODULE ON THE FOUR BENCHMARKS. THESE RESULTS DIRECTLY EFFECT THE OVERALL PERFORMANCE OF THE PROPOSED FRAMEWORK.

| Benchmark          | RSE Accuracy (%) | RFDB Accuracy (%) |
|--------------------|------------------|-------------------|
| UCF-QNRF           | 87.8             | 93.0              |
| ShanghaiTech       | 92.0             | 91.4              |
| WorldExpo'10       | 91.7             | 94.2              |
| AHU-Crowd          | 89.2             | 92.3              |

TABLE XI
ZOOMCOUNT PERFORMANCE ANALYSIS ON SHANGHAItech AND UCF-QNRF BENCHMARKS USING DIFFERENT ML CLASSIFICATION ALGORITHMS IN THE RFDB BLOCK OF DECISION MODULE (DM). AS SHOWN, TOP FIVE RESULTS INDICATE BEST PERFORMANCE BY THE RANDOM FOREST ALGORITHM, THUS, JUSTIFYING ITS USAGE IN THE RFDB MODULE.

| Algorithm            | ShanghaiTech | UCF-QNRF |
|----------------------|--------------|----------|
| Regression           | MAE [0.190]  | RMSE [97.5] |
| ExtraTrees           | 70.7         | 102.8    |
| GradientBoosting     | 72.9         | 119.0    |
| AdaBoost             | 75.0         | 105.31   |
| Logistic Regression  | 78.9         | 119.6    |

The classification accuracy of the decision module directly affects the overall performance of the proposed method. In this section, we show the individual classification accuracy performance of the RSE and RFDB based DM modules on the four benchmarks. The accuracy results are shown in Table X from which we can see that both modules perform quite effectively. Also, the RFDB version of the DM module gives slightly better accuracy in most cases compared to the RSE based DM module. The overall system outperforms the state-of-the-art owing to the improved and reasonable classification accuracy of the DM module.

G. RFDB Algorithm Selection

In this paper, we adopt the Random Forest algorithm for the RFDB module. In practice, other machine learning-based classification algorithms can also be employed. In order to choose the best one for our system, we experimented with different classifiers to select the appropriate decision-making algorithm. The results based on the ShanghaiTech and UCF-QNRF datasets are shown in Table XI. We observe that ensemble based methods perform better on our relatively smaller and imbalanced RFDB datasets as they prevent over-fitting and high variance by combining several machine learning techniques. After evaluation, the Random Forest appears to be the best choice as the RFDB algorithm as shown in Table XI where the top five best results justify the selection of the Random Forest algorithm. For these experiments, we used machine learning library scikit-learn for python programming [2].

H. Qualitative Results

In Fig. [10], we show some good and bad case qualitative results from UCF-QNRF and ShanghaiTech datasets. We also compare our results with the ground truth (GT), DenseNet [20] Regression (DR) and the state-of-the-art density map methods. In each row, the first three cases demonstrate the good results followed by two bad estimates. The bad case results happen mostly due to the test image being detected as wrong extreme case type by the decision module (DM). We also show some crowd-density classification visual results to demonstrate the qualitative performance of the CDC classifier in Fig. [11].

I. Computational Time Analysis

In this experiment, we compare the proposed network with two state-of-the-art models (CSRNET [26], CP-CNN [40]) in terms of real computational time on the ShanghaiTech [14] dataset. The results are shown in Table XII, where \( T_{total} \) and \( T_{avg} \) represent the total and average time taken for the whole dataset test images respectively, whereas, \( T_{smallest} \) and \( T_{largest} \) represent the computation time taken for the smallest and the highest resolution test image respectively in the dataset. For the fairness of comparison, all networks have been evaluated on the same NVIDIA Titan Xp GPU.

As shown in the Table XII our method takes a reasonable computation time (in between the two state-of-the-art networks with much closer to the one with the lesser time). However, the proposed method outperforms other models on all standard evaluation metrics. Since our method is modular, we have also shown the pipeline approach (ZoomCount − RSE_{pt}, ZoomCount − RFDB_{pt}) on the same test images, where next image patches start using the CDC classifier once previous image patches are done using the classifier and now passing through the next modules (the Decision Module and the CRM
Fig. 10. Some examples of the qualitative results. First and second rows show qualitative results of our method on the UCF-QNRF [22] and ShanghaiTech [49] datasets respectively. First three columns show good results, followed by two bad case images. Each result also shows the estimates of DenseNet [20] Direct Regression (DR) and the Density map method as a comparison.

TABLE XII

| Method                | $T_{total}$ (secs) | $T_{avg}$ (secs) | $T_{smallest}$ (secs) | $T_{largest}$ (secs) | MAE   | RMSE  |
|-----------------------|--------------------|------------------|-----------------------|----------------------|-------|-------|
| CSRNET [26]           | 60.2               | 0.33             | 0.09                  | 0.42                 | 68.2  | 115.0 |
| CP-CNN [40]           | 122.8              | 0.68             | 0.31                  | 0.85                 | 73.6  | 106.4 |
| ZoomCount-RSE         | 85.4               | 0.47             | 0.14                  | 0.57                 | 66.6  | 94.5  |
| ZoomCount-RFDB        | 89.4               | 0.49             | 0.15                  | 0.59                 | 66.0  | 97.5  |
| ZoomCount – RSE$_{pl}$| 61.9               | 0.34             | -                     | -                    | 66.6  | 94.5  |
| ZoomCount – RFDB$_{pl}$| 63.5               | 0.35             | -                     | -                    | 66.0  | 97.5  |

VI. CONCLUSION

In this work, we have proposed a novel zoom-in and zoom-out based mechanism for effective and accurate crowd counting in highly diverse images. We propose to employ a decision module to detect the extreme high and low dense cases, where most state-of-the-art regression and density map based methods perform worse. The cluttered background regions are also discarded using the rigorous deep CNN 4-way classifier. Even without using any density maps, ZoomCount outperforms the state-of-the-art approaches on four benchmark datasets, thus proving the effectiveness of the proposed model.

REFERENCES

[1] “Scikit sklearn library,” https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. [Accessed: April 13, 2019].
[2] “Scikit sklearn library,” https://scikit-learn.org/stable/supervised_learning.html. [Accessed: April 27, 2019].
[3] “UCF-QNRF online demo,” http://crcv.ucf.edu/crowd_counting.php. [Accessed: November 11, 2018].
[4] D. Babu Sam, N. N. Sajjan, R. Venkatesh Babu, and M. Srinivasan, “Divide and Grow: Capturing Huge Diversity in Crowd Images With Incrementally Growing CNN,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3618–3626.
[5] V. Badrinarayanan, A. Kendall, and R. Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for image segmentation,” IEEE transactions on pattern analysis and machine intelligence, vol. 39, no. 12, pp. 2481–2495, 2017.
[6] 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), 2018, pp. 734–750.
