AECNet: Attentive EfficientNet For Crowd Counting

Muskan Dosi, Kartik Thakral, Surbhi Mittal, Mayank Vatsa and Richa Singh
Indian Institute of Technology, Jodhpur

***Abstract*— In the COVID pandemic situation, crowd counting became one of the tools to monitor if the social-distancing norms are being followed or not. However, in designing crowd counting algorithm, there are several challenges such as background noise, camera-to-objects distance, occlusion, and variations due to illumination, scale, and viewpoint. In this research, we propose a novel pipeline for density estimation in crowd counting. The proposed pipeline makes use of an encoder-decoder-based architecture in which we explore the family of EfficientNets for the encoder architecture. For the decoder, we propose a deeper attention network to assist the model in a better distinction between foreground and background pixels. We empirically show that for a crowd counting dataset, the use of average pooling operation for any backbone architecture of encoder gives a significant improvement in performance. In terms of Mean Absolute Error, the proposed pipeline outperforms existing state-of-the-art techniques by a large margin on large-scale and small-scale counting datasets, UCF-QNRF and UCF_CC_50 dataset. We also achieve state-of-the-art results on the ShanghaiTech and Mall datasets. We additionally propose a crowd counting dataset captured using drones. We perform benchmark experiments on this dataset with existing and the proposed methods.The proposed dataset can be found at http://www.iab-rubric.org/resources/CrowdUAV.html.**

I. Introduction

Counting the number of people in a crowd through an image or video is a tedious and challenging task in computer vision. This is due to the occlusion of heads in the image, scale variation due to distance from the capturing device, variation in the orientation of capturing the input, and variation in illumination conditions. Some of these challenges have been illustrated in Fig. 1. The early work on crowd counting through deep learning includes head detection [23], [34] in which the count is estimated through detecting the heads in a crowd. However, such techniques fail in densely crowded regions. Other techniques for count estimation is direct count regression [22] where image features are fed to fully-connected regression layers or to a regression support vector machine [22]. These regression models directly predict the count of people in an input image. Such techniques generally lack explainability and do not generalize well in real-world scenarios.

The recent trend in crowd counting through aggregating density maps still prevails and predominates other algorithms. The technique of count estimation through density maps was first introduced by Lempitsky Zissermann[13] and has become the de-facto approach for crowd counting. For a given annotation map, a density map is generated using the convolution of original labels with a Gaussian kernel, the sum of which results in the crowd count. Recent works

978-1-6654-3176-7/21/$31.00 ©2021 IEEE

Fig. 1. Challenges observed in crowd counting datasets.

for the crowd counting problem emphasize designing multi-column architectures [2], [33] and incorporating attention mechanisms [21], [32] to reduce the influence of background information and manage attention towards the crowd proportional to its density. Segmentation maps generated through point annotations are also being used for adding attention to networks [30].

In this research, we propose a novel encoder-decoder-based pipeline with a combination of attention and regression networks for crowd counting via density map generation. In the encoder, we show the effectiveness of average-pooling layers with the chosen backbone architecture. The feature maps from the encoder are fed to a decoder network and a regression network. The decoder network consists of an attention sub-network and density map generator sub-network. Since the crowd datasets contain a large amount of background noise, the attention network in the decoder forces the model to focus on relevant foreground features only. For training, the loss from the regression network is added to the loss obtained from the generated segmentation and density maps.

We also propose a novel dataset called CrowdUAV for crowd counting. The CrowdUAV dataset is captured using a drone or Unmanned Aerial Vehicle (UAV) from various altitudes and viewpoints under diverse illumination conditions. Recently, a large-scale drone-based database DroneCrowd was introduced [31]. It contains videos captured at different illumination, scale, and densities. While the database has extensive annotations for various tasks such as object detection, tracking, and counting, it is fairly limited in image viewpoint. In the proposed dataset, we cover various viewpoints and

at a much lower altitude. The crowd density varies from a minimum of 3 people per frame to a maximum of 191 people per frame. We provide benchmark results using existing and proposed algorithms for the dataset.

II. Related Work

Various approaches have been proposed for crowd counting in the literature such as detection-based, regression-based, convolutional neural networks, density map estimation and attention-based mechanisms.

Detection-based techniques Early works on crowd counting are based on estimating count through detection of different body parts of a person [5]. Object detectors such as RCNN ,YOLO and SSD were used for detection. This was efficient only for low density images and could not perform well on dense crowd images. Bounding box annotation for such techniques is also a time consuming task.

Regression-based techniques Regression-based techniques directly estimate the count of people from a patch of image. This method involves extracting global [4] and local [19] features from images, and then uses some regression function to learn the image to count mapping.

Density map estimation techniques Due to the limitations of regression based techniques, density map based crowd counting techniques were introduced [13].

CNN-based techniques The density map estimation was improved using Convolutional Neural Networks (CNNs) [6]. Initial work used basic CNN models for density map generation [27], [29]. Multi-column network architectures usually adopt different columns to capture multi-scale information corresponding to different receptive fields which worked well in handling high scale variation in images. MCNN [33], Switching-CNN [1], DadNEt [8] and CrowdNet [2] are some examples of Multi-column architecture. Deeper CNNs are also used in crowd counting [3], [14] .

Multi-task techniques Multi-task architectures employing multi-task learning are also popular which combine density generation with detection and segmentation. Some of these multi-task architectures are DecideNet [15], PCCNet [7].

Attention-based techniques Attention networks using segmentation maps are also being actively used for crowd counting, to focus on important features of images. Some models use it for detecting heads in the image [32], while some are used for foreground-background separation in crowded images. Scale variation is also handled by different attention models [12], [26].

III. Proposed AECNet Approach

In the proposed network, we make use of an encoder-decoder architecture with a regression neural network. We start by addressing the problem of scale variation in the crowd density estimation datasets by taking the 224×224 size RGB image from a mini-batch of size 24. We then generate two scaled versions of the same image. As image preprocessing steps, we augment the images in the mini-batch by horizontal mirroring, varying the contrast and rotating the image by 5^0 in both clockwise and anti-clockwise direction followed by Gaussian smoothing. The input image is divided into four patches which are given as input to the model. These input images are fed to the EfficientNet [24] based encoder. The feature maps from this encoder are fed to a decoder architecture and a regression neural network. Fig. 2 illustrates the entire pipeline of proposed EACNet and the algorithms are explained in the following sections.

A. Encoder

Deep Convolutional Neural Networks have been used as encoders to extract features from crowd images in the literature. These features are then consumed to generate crowd predictions. Various backbone models have been explored to extract features from crowd images like VGG16 [8], Inception-V3 [30] among others. This paper makes the first attempt to use the family of EfficientNets [24] as the backbone model due to its high accuracy and better efficiency over existing CNNs. Since the datasets for crowd counting are generally small (due to tedious annotation), we use EfficientNet-B0 or EfficientNet-B1 as the backbone feature extractor to prevent overfitting.

Generally, the crowd datasets are point annotated representing the head of a person in the given image. We observe that the heads in the images for crowd counting are usually of relatively darker pixels. Inspired by this observation, we propose to use average pooling layers. Since max-pooling layers propagate brighter pixels better, they are less likely to provide important information represented in head pixels. With this notion, we add some average pooling layers in the encoder of the proposed network. This transmits the pixels indicating heads which provides for better feature representation. This effect is further studied in Section V-C. The features $\boldsymbol{F}^{en}$ from the encoder are then utilized by the decoder network and the regression neural network.

B. Decoder

The decoder network utilizes the features $\boldsymbol{F}^{en}$ from the encoder network to output the segmentation map $\boldsymbol{M}^s \in \{0,1\}^{W\times H}$ and density map $\boldsymbol{M}^d \in \mathbb{R}^{W\times H}$ where W and H are the width and height of the image. We begin with using Pixel Shuffling [20] for upsampling the feature map which rearranges the pixels to obtain features with higher dimensions. This operation is performed to reduce the risk of error probability caused by bilinear interpolation. After pixel shuffling, we employ the Transpose Convolution layers (TConv layers) in the decoder network. The output from TConv layers is then passed onto two modules: 1) the Attention Network (AN) and 2) the Density Map Generator (DMG). The Attention Network is used to generate the segmentation map corresponding to a given image and is used for foreground-background separation. The Density Map Generator outputs the density map.

Attention Network: We employ a deeper attention network consisting of two transpose convolution-based attention layers followed by a sigmoid activation. A max-pool layer is sandwiched between the two attention layers to propagate

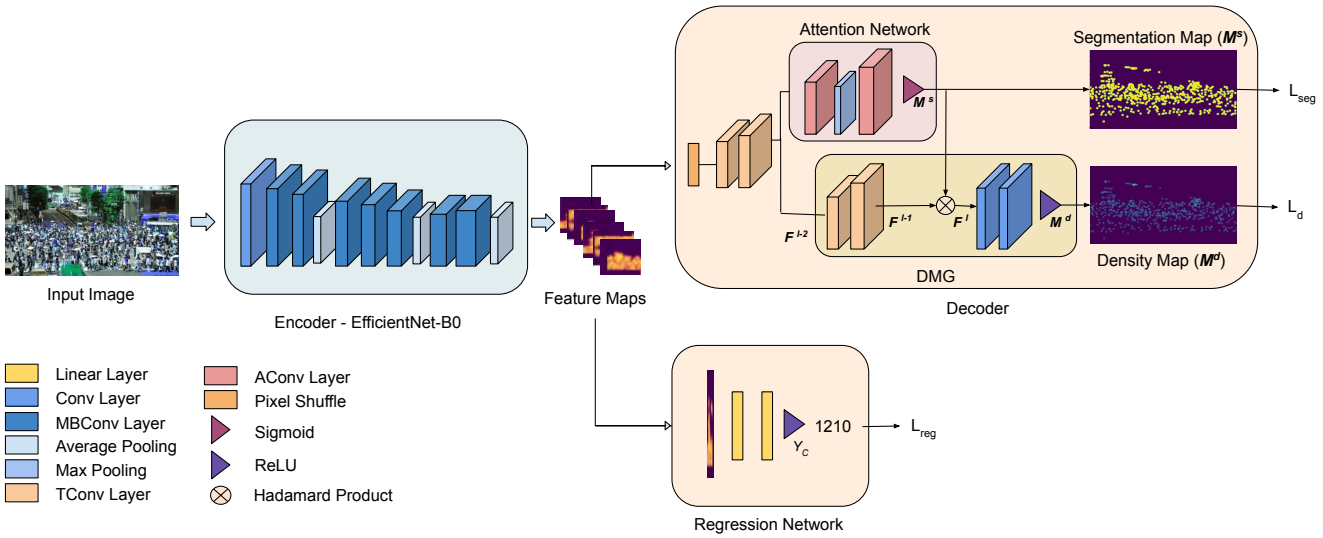

Fig. 2. The steps involved in the proposed AECNet. AECNet uses scaling in image space and EfficientNet-B0 with average pooling layers as the backbone for the encoder network. The extracted features from this encoder network are given to 1) the decoder network. The decoder uses pixel shuffling, transpose convolution, Attention Network (AN), and Density Map Generator Network (DMG) and outputs the segmentation map and the density map. 2) A regression neural network that predicts the head count in the input image. We combine the losses from both networks to train the entire network.

only the most sensitive information to the next attention layer. Here each transpose convolution layer is followed by a 1×1 convolution operation and we term it as the attention convolution layer (**AConv** layer). For this, we give the feature maps from the second last decoder layer $\boldsymbol{F}^{l-2}$ into the attention network. We employ the segmentation map as an attention map to focus on the foreground consisting of point-annotated heads while ignoring the background region. These generated segmentation maps are then compared with the ground-truth segmentation maps, and the Binary Cross-Entropy Loss L_{attn} is calculated. The attention network is partly inspired by the work of [30].

$$\boldsymbol{M}^s = \sigma(((\boldsymbol{F}^{l-2} \circledast \textbf{AConv}_1) \circledast \textbf{Maxpool}) \circledast \textbf{AConv}_2) \quad (1)$$

This output generated from the attention network is the segmentation map of the given image. Next, we take the Hadamard product of the output segmentation map $\boldsymbol{M}^s$ with the feature maps $\mathbf{F}^{l-1}$, i.e., the feature maps from the last TConv layer and give it as an input to the DMG module.

$$\boldsymbol{F}^l = \boldsymbol{M}^s \odot \boldsymbol{F}^{l-1} \quad (2)$$

The segmentation map generated from the attention network has pixel values of 1 where the person can be identified and 0 elsewhere. This brings attention in the final feature map by magnifying the weights of the locations where persons exist in the crowd. In other words, this attention mechanism forces the model to focus more on the foreground and neglect the background region in the image.

Density Map Generator (DMG): The other module in the decoder network is the DMG which outputs the final density map $\boldsymbol{M}^d$. The summation over this density map is calculated to estimate the count of people in the input image. For this, we make use of two convolutional layers which input the attention-weighted feature map $\boldsymbol{F}^l$. Here, the density loss L_d is computed as the L2 loss between the predicted density map and the ground truth density map for training.

Regression Network: The regression neural network is a multilayer perceptron that uses the flattened features from the encoder and predicts a single value greater than or equal to zero. The predicted value is the estimation of people in the crowd, and we calculate the L2 loss L_{reg} between the predicted value and the ground truth value.

C. Overall Loss

In the crowd counting datasets, the information corresponding to the crowd image is the point head annotation map $\hat{\boldsymbol{Y}}_M \in \mathbb{R}^{W \times H}$ and the total number of people $\hat{Y}_C \in \mathbb{Z}$ in the crowd image. Here, the crowd counting problem is formulated as a regression of the density map. We generate the ground truth density map and segmentation map from the available point head annotations. For the generation of ground truth segmentation maps $\hat{\boldsymbol{M}}^s$ and density maps $\hat{\boldsymbol{M}}^d$, we refer to [33]. We use a box matrix $B_n(x)$ of ones of size $n \times n$, centered at x for generation of segmentation maps and Gaussian matrix G_σ for the generation of density maps.

$$\hat{\boldsymbol{M}}^s(x) = \sum_{i=1}^{N} \hat{\boldsymbol{Y}}_M(x - x_i) \circledast B_n(x) \quad (3)$$

$$\hat{\boldsymbol{M}}^d(x) = \sum_{i=1}^{N} \hat{\boldsymbol{Y}}_M(x - x_i) \circledast G_\sigma(x) \quad (4)$$

In the experiments, we set $n = 25$ as the size of box matrix and $\sigma = 4$ for the variance of Gaussian filter.

We enforce the binary cross-entropy loss between predicted segmentation maps $\boldsymbol{M}_s$ and the ground truth segmentation maps $\hat{\boldsymbol{M}}_s$ and call it Attention loss L_{attn}:

$$L_{attn} = -\frac{1}{N}\sum_{i=1}^{N}[\boldsymbol{M}_i^s \odot log(\hat{\boldsymbol{M}}_i^s) + (1 - \boldsymbol{M}_i^s) \odot log(1 - \hat{\boldsymbol{M}}_i^s)]$$

From the DMG module of the decoder, we calculate the L2 loss (referred as the density loss L_d) between the predicted density map $\boldsymbol{M}_d$ and the ground truth density map $\hat{\boldsymbol{M}}_d$:

$$L_d = -\frac{1}{2N}\sum_{i=1}^{N}||\boldsymbol{M}_i^d - \hat{\boldsymbol{M}}_i^d||_2^2 \tag{5}$$

In parallel, the Mean Squared loss between the ground truth count $\hat{Y}_C$ and the value Y_C predicted from the regression neural network is calculated. This loss is termed as the regression loss L_{reg}, given by:

$$L_{reg} = -\frac{1}{2N}\sum_{i=1}^{N}||Y_{C,i} - \hat{Y}_{C,i}||_2^2 \tag{6}$$

The total loss L_{total} is calculated as the weighted sum of the above three losses and the loss is backpropagated through the entire network.

$$L_{total} = \lambda_1 L_{attn} + \lambda_2 L_d + \lambda_3 L_{reg} \tag{7}$$

where λ_1, λ_2 and λ_3 are the hyperparameters. In the experiment, the value of λ_1 is set to be 100, λ_2 is 10 and, λ_3 is 0.5.

Implementation Details: We perform patch-wise training by dividing an image into four equal parts. For each patch of an image, we generate multiple augmentations followed by Gaussian smoothing. For data augmentation, contrast variation, horizontal flipping at 0.5 probability, and rotation at 5 degrees is performed, and the variance is set to four for smoothing. EfficientNet-B0 and EfficientNet-B1 pre-trained on the ImageNet dataset are used as backbone networks for extracting features. We add three average pooling layers in the backbone network after the 3^{rd}, 6^{th} and, 8^{th} MBConvBlocks. We also replace these layers with max-pooling for experiments in Section V-C. The network is trained using Adam optimizer with an initial learning rate of 1e-4 for 200 epochs, and the learning rate is decayed after every 20 epochs.

IV. Datasets and Evaluation Metrics

The results of the proposed algorithm have been evaluated on several existing datasets and the proposed CrowdUAV dataset. This section summarizes the details and protocols of the databases.

A. Existing Databases

We have used four existing databases for evaluation.
ShanghaiTech [33] is one of the largest large-scale crowd counting datasets in recent few years which consists of 1198 images with 330,165 annotations. It is divided into two parts i.e, PartA and PartB. PartA has images with high crowd density and consists of 300 training images and 182 testing images, whereas PartB has 400 training images and 316 testing images with relatively lower density.
UCF_CC_50 [9] contains very high density 50 crowd images which makes it a very challenging dataset. It was created by scraping from publicly available web images. It is enriched with high perspective and illumination variation. Since the dataset contains only 50 images, a 5-fold-cross-validation is performed for training and testing.
UCF_QNRF [10] is another challenging dataset, which contains 1535 images with large scale variation and The number of people in images ranges from 49 to 12,865. In this dataset, images have different scenes and a wide variety of viewpoints, densities, and illumination variations. 1201 images are used for training, and the remaining 334 images are used for testing.
Mall dataset [4] is collected from a surveillance video from a shopping mall, containing 2000 video frames having 62,325 annotations in total. We use 800 images for training, and the remaining 1200 images are used for testing.

B. Proposed CrowdUAV Database

The CrowdUAV dataset is captured using a drone or Unmanned Aerial Vehicle (UAV). The dataset contains 500 images with a maximum of 191 people and a minimum of 3 people in a single frame. The dataset offers variations across pose, illumination, occlusion, and varying degrees of density in images. Some samples from the database can be seen in Fig. 4. In the CrowdUAV dataset, the images have been captured at a much lower altitude compared to the DroneCrowd dataset [31] and have a larger variation in viewpoint. The dataset will be publicly released to the research community.

Data Collection and Annotation: The data is collected as videos using the DJI Phantom-4 drone, which is a high-end, entry-level professional drone. The videos have been captured at a frame rate of 30fps, and at a resolution of 720p. From the collected videos, frames are extracted at a rate of 30fps. Each of the frames is of dimension 3840×2160. After removing the redundant and blurry frames, we perform experiments on a total of 500 frames. These frames are annotated by the point annotation method using the online VGG annotator tool[1]. The annotations are saved in MAT format for all the images.

Dataset Statistics and Protocol: The proposed CrowdUAV dataset contains a total of 500 images. The dataset has been partitioned into two disjoint sets. The training set contains

[1]https://www.robots.ox.ac.uk/ vgg/software/via/via.html

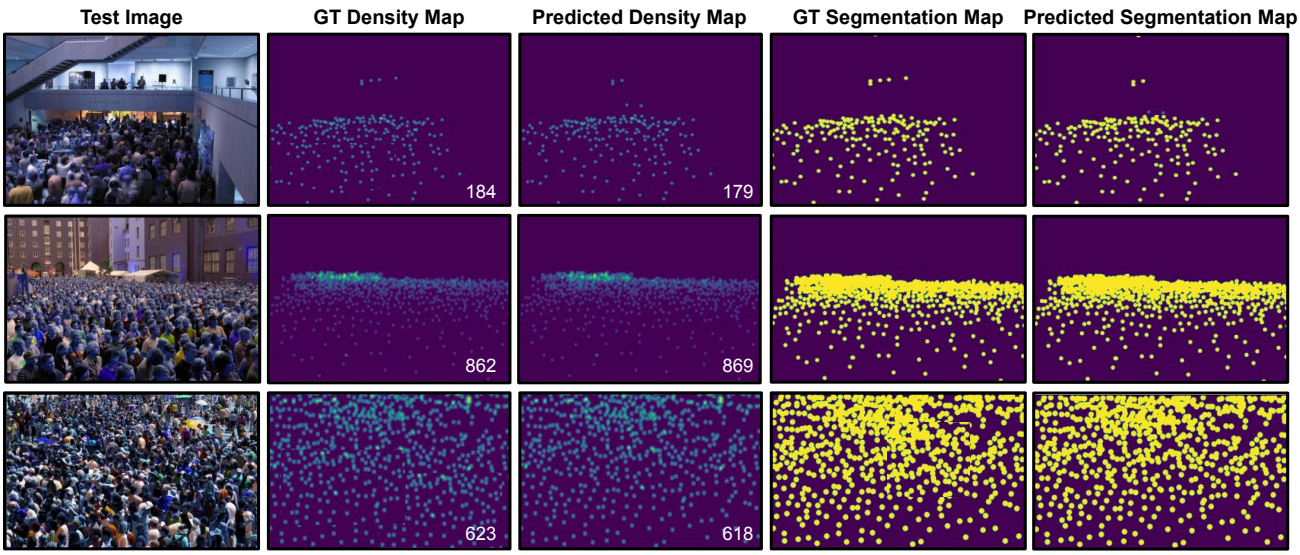

Fig. 3. Visualization of density maps and segmentation maps on images from different datasets. First row is a sample from ShanghaiTech Part-A, second row contains sample from ShanghaiTech Part-B and third row sample is taken from UCF_CC_50 dataset. First column is the input test image, second column is the corresponding ground truth density map and, the third column is the density map predicted by our pipeline. Similarly, the second-last column is the ground truth segmentation map and the last column is the predicted segmentation map.

Fig. 4. A snapshot of samples from the proposed CrowdUAV dataset.

70% of the total images i.e., 350 images, while the testing set contains the remaining 150 images. The training and test set contains a balanced combination of crowd densities, illumination, and viewpoint variations. Table I provides the statistics of the proposed dataset along with existing crowd counting datasets.

SSIM and PSNR metrics are used to evaluate the quality of generated density map. Crowd count is calculated by taking a summation over all the pixels of the density map. Following [18], [28], [30], Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) are computed on the test set to evaluate the performance of different models based on the predicted and true crowd count.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|Y_{C,i} - \hat{Y}_{C,i}| \tag{8}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Y_{C,i} - \hat{Y}_{C,i})^2} \tag{9}$$

TABLE I

DATASETS AND THEIR STATISTICS. COLS 1-2 DESCRIBE THE NAME OF THE DATASET AND THE NUMBER OF IMAGES IN EACH DATASET, RESPECTIVELY. COLS 3-5 DESCRIBE THE TOTAL, MINIMUM AND MAXIMUM ANNOTATIONS, RESPECTIVELY, AVAILABLE IN THE DATASET.

Dataset	No. of Images	Total	Min	Max
Mall [4]	2,000	62,325	13	53
UCF_CC_50 [9]	50	63,974	94	4,543
UCF_QNRF [10]	1535	1,251,642	49	12,865
SHT_A [33]	482	241,677	33	3,139
SHT_B [33]	716	88,488	9	578
CrowdUAV (Proposed)	500	44,542	3	191

V. EXPERIMENTS AND RESULTS

The performance of the proposed approach is tested extensively on multiple databases, and the performance is compared with several state-of-the-art approaches. The results are shown in Table III.

A. Comparison with state-of-the-art

The experiments have been performed on four benchmark datasets - ShanghaiTech [33], UCF_CC_50 [9], UCF_QNRF [10], and Mall dataset [4]. The results of the proposed and existing algorithms have been summarized in Table III. It is evident that the proposed pipeline outperforms all the existing frameworks, including the state-of-art MAE on all datasets, with the best RMSE on ShanghaiTech PartB and UCF_QRNF, and the second-best on UCF_CC_50 and Mall datasets. Especially for UCF_CC_50 dataset, we achieve significantly better MAE. With improvement in performance, we observe that the proposed pipeline converges relatively faster and to a lower loss value. Essentially, this happens because the EfficientNet backbone uses a relatively smaller number of parameters to train and hence, converges faster.

TABLE II

SSIM AND PSNR COMPARISONS ON SHANGHAITECH-A DATASET.

Method	SSIM	PSNR
MCNN [33]	0.52	21.40
CSRNet [14]	0.76	23.79
TedNet [11]	0.83	25.88
CFANet [18]	0.88	30.11
AECNet	**0.91**	**30.12**

As shown in Fig. 5, our pipeline achieves the lowest MAE at 60 epochs, whereas the convergence for other techniques starts after 80 to 120 epochs.

We also compare the quality of the density maps obtained using mean SSIM and mean PSNR values. These values are calculated on the ShanghaiTech PartA dataset and are available in Table II. The results suggest that the proposed pipeline generates the best quality density maps.

From Fig. 3, we illustrate the ground truth density and segmentation map and compare it with the maps generated through the proposed pipeline. The visualization shows that the distribution of crowd in the predicted density and segmentation maps is very close to the ground truth maps.

Benchmark results on CrowdUAV: The results on CrowdUAV using the various existing algorithms(MCNN[33], CSRNet[14], CFANet[18] and SGANet[30]) and the proposed algorithms are summarized in Table IV. For comparison, we chose recent deep learning techniques whose codes are publicly available. The proposed algorithm provides the best results among all the other algorithms that have been recently proposed.

B. Effect of Backbone Architecture

We also empirically study the effect of different backbone networks in the encoder for feature extraction. For this, we experiment with multiple models from VGG, ResNet, Inception, and EfficientNet families on the ShanghaiTech PartB dataset. The results are presented in the supplementary. The results suggest that EfficientNets perform best. We prefer the EfficientNet-B0 baseline model because the crowd counting datasets are generally small, and a deeper model generally overfits.

C. Effect of Average Pooling

Most of the off-the-shelf CNN architectures uses max-pooling layers. In crowd counting datasets, heads generally acquire relatively darker pixels. This paper hypothesize that it is difficult for the feature extractor to generate the activation from darker pixels and propagate it through a max-pooling operation. We first train the proposed pipeline to validate the idea by adding three max-pool layers to the backbone for the ShanghaiTech PartA dataset. Next, we train the same pipeline but with average pooling layers. After convergence of both the models, we visualize the feature maps obtained from the last pooling layer (i.e, pooling layer after MBConvBlock 8) in Fig. 6. For the visualization, we choose samples such that the crowd regression scores predicted from the model with average-pool and max-pool are close. We then aggregate all

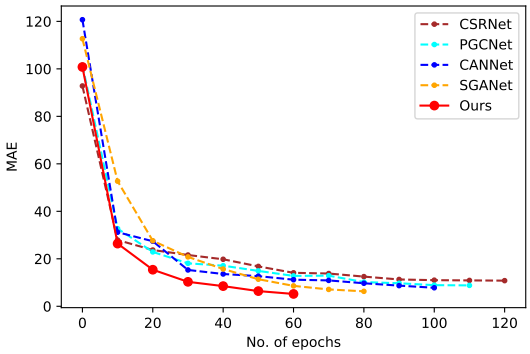

Fig. 5. **The convergence graph of different techniques.** The proposed pipeline with EfficientNet-B0 backbone converges significantly faster than existing algorithms. It is also able to achieve a lower Mean Absolute Error (MAE) for ShanghaiTech-B dataset.

the feature maps, upsample them and superimpose it over the input image. It can be seen that both the models predict the crowd count to be very close to the ground truth. However, the output from the average pooling correctly focuses only on the heads of each person. In contrast, in the case of max-pooling, the activations are propagated from areas other than the head region and the background regions.

To study the efficacy of using average pooling, we perform the experiment by replacing all the max-pool layers of existing deep learning-based crowd counting techniques[1] with average-pool layers. The results are summarized in Table V. They depict a gain in performance of each of the techniques with average pooling layers. We achieve lower MAE and RMSE values from all the existing techniques on replacing all the max-pool layers with average-pool layers.

The observations from these experiments validate our hypothesis. We propose to prefer the average-pooling operation instead of the max-pooling operation in the encoder backbone for the problem of crowd counting.

D. Ablation Study

In this section, we study the impact of the different components in the pipeline of the proposed AECNet. We conduct the ablation study on ShanghaiTech Part B dataset and analyze the impact of different components of the pipeline on the performance. Our observations are summarized in Table VI. We begin by training a simple encoder-decoder architecture with EfficientNet-B0 as encode. For decoder architecture, we add pixel-shuffling (PS) and observe a gain in performance. This gain is observed because pixel shuffling enables better feature reconstruction. We then use the Attention Network (AN) in the decoder which leads to a significant performance gain. The gain in the performance is because AN guides the model to focus more on the foreground region and ignore the background region. We also test this model with a regression network. After this, we add max-pool and average-pool

[1]We chose recent deep learning techniques whose codes are publicly available.

TABLE III

COMPARISON OF RESULTS OBTAINED FROM THE PROPOSED PIPELINE WITH EXISTING ALGORITHMS. THE BEST PERFORMANCE ON EACH DATASET IS HIGHLIGHTED BY **BOLD** AND SECOND BEST IS HIGHLIGHTED BY UNDERLINED. IN THE TABLE, EFFICIENTNET-B0 IS USED AS THE BACKBONE ARCHITECTURE FOR SHANGHAITECH AND UCF_CC_50 DATASETS, WHILE EFFICIENTNET-B1 IS USED FOR UCF_QRNF AND MALL DATASET.

	ShanghaiTechA		**ShanghaiTechB**		**UCF_CC_50**		**UCF_QRNF**		**Mall**	
Algorithms	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
MCNN [33]	110.2	173.2	26.4	41.3	377.6	509.1	277	426	-	-
DRSAN [16]	69.3	96.4	11.1	18.2	219.2	250.2	-	-	1.72	2.1
CSRNet [14]	68.2	115	10.6	16	266.1	397.5	-	-	-	-
SANet [3]	67	104.5	8.4	13.6	258.4	334.9	-	-	-	-
DADNet [8]	64.2	99.9	8.8	13.5	285.5	389.7	113.2	189.4	-	-
DecideNet [15]	-	-	20.75	29.42	-	-	-	-	1.50	1.92
CANNEt [17]	62.3	100	7.8	12.2	212.2	**243.7**	107	183	-	-
SGANet [30]	57.6	100.4	6.3	9.8	221.9	289.8	-	-	-	-
M-SFANet [25]	57.55	94.48	6.32	10.06	162.33	276.76	85.60	147.78	-	-
DM-count [28]	59.7	95.7	7.4	11.8	211	291.5	85.6	148.3	-	-
CFANet [18]	56.1	**89.6**	6.5	10.2	203.6	287.3	89	152.3	1.20	**1.56**
AECNet	**55.2**	92.4	**5.12**	**9.6**	**152.78**	251.6	**84.2**	**142.7**	**1.17**	1.62

TABLE IV

BASELINE RESULTS ON THE PROPOSED CROWDUAV DATASET.

Method	**MAE**	**RMSE**
MCNN [33]	182.6	245.8
CSRNet [14]	142.8	210.0
CFANet [18]	94.1	152.8
SGANet [30]	97.4	163.2
AECNet (Proposed)	**75.7**	**116.9**

TABLE V

COMPARISON OF DIFFERENT ALGORITHMS WITH AND WITHOUT AVERAGE POOL ON SHANGHAITECH-A DATASET.COLS 2-3 SHOW THE TESTING PERFORMANCE OF DIFFERENT TECHNIQUES WITH THE DEFAULT SETTING (I.E., WITH MAX-POOLING OPERATION). THE EXPERIMENTS WITH AVERAGE-POOLING (COLS 4-5) SHOW THE TESTING PERFORMANCE AFTER REPLACING ALL THE MAX-POOL LAYERS WITH AVERAGE-POOLING IN THE ENCODER MODEL.

	Without Avg Pooling		**With Avg Pooling**	
Method	**MAE**	**RMSE**	**MAE**	**RMSE**
CSRNet[14]	68.2	115	62.7	112.4
CANNet[17]	62.3	100	61.6	99.9
SGANet[30]	57.6	100.4	58.61	100.2
AECNet	62.8	98.8	55.2	93.4

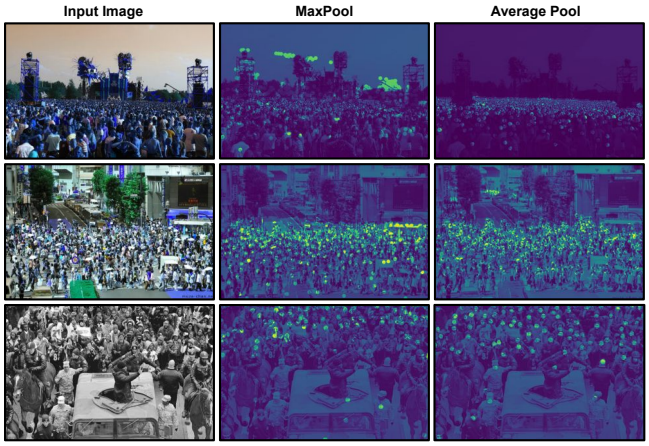

Fig. 6. **Effect on average pool on feature maps.** First column represents the test image, second column represents the feature map extracted from last max-pool layer, and the third row represents the feature map from the last average pool layer of the proposed model.

layers to the encoder and check the performance individually. We notice the performance with average-pool is better than max-pool. We then test the performance after using both pixel-shuffle and attention network and notice a significant performance gain compared to the initial encoder-decoder architecture. We then use average pooling in the encoder and the pixel-shuffle with attention network in the decoder. With this pipeline, we achieve the state-of-the-art results on the ShanghaiTech PartB dataset. Finally, we train the entire pipeline with a regression network that consumes the features extracted from the encoder and regress the crowd count. The loss from this network is added to the total loss for the training of the entire pipeline and it pushes the performance even further. We also study the impact of different backbone architectures using the proposed pipeline, and the results are made available in the supplementary text.

VI. CONCLUSION

Crowd counting has several applications, particularly in the pandemic situation. This is an arduous research problem when the camera is at a large stand-off distance and there are multiple occluded individuals in a given frame. To address this research challenge, we propose a novel encoder-decoder-based pipeline for crowd counting. In the encoder block, we use EfficientNets, and empirically show that they should be preferred over other feature extraction architectures. We also explore the use of average-pooling over max-pooling and observe that for the problem of crowd counting, average-pooling should be preferred in the encoder block. We also build over existing attention mechanisms and propose a deeper attention network while maintaining the dimensions of the feature maps. We additionally append a regression

TABLE VI

ABLATION STUDY W.R.T TO EACH MODULE OF THE PIPELINE ON SHANGHAITECH-B DATASET WITH EFFICIENTNET-B0 BACKBONE. (PS: PIXEL SHUFFLING, AN: ATTENTION NETWORK, AVG POOL AND MAX POOL: THREE MAX-POOL LAYERS AND AVERAGE-POOL LAYERS IN ENCODER RESPECTIVELY, AND RN: REGRESSION NETWORK.

Components Present in the Pipeline	**MAE**	**RMSE**
EfficientNet	13.05	26.34
EfficientNet + PS	11.75	20.12
EfficientNet + AN	10.45	16.87
EfficientNet + RN	12.8	23.2
EfficientNet + Max Pool	12.8	24.6
EfficientNet + Avg Pool	11.9	22.8
EfficientNet + PS +AN	9.45	14
EfficientNet + PS + AN + Max Pool	7.0	11.5
EfficientNet + PS + AN + Avg Pool	5.25	10.2
EfficientNet + PS + AN + Max Pool + RN	6.72	11.6
EfficientNet + PS + AN + Avg Pool + RN	5.12	9.6

neural network with the encoder-decoder architecture and enhance the state-of-the-art results. This paper also presents a crowd counting-based dataset which is captured through a drone along with benchmark results.

VII. ACKNOWLEDGEMENTS

K. Thakral is partly supported by the PMRF Fellowship. S. Mittal is partially supported by the UGC-Net JRF Fellowship. M. Vatsa is partially supported through the Swarnajayanti Fellowship.

REFERENCES

[1] D. Babu Sam, S. Surya, and R. Venkatesh Babu. Switching convolutional neural network for crowd counting. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5744–5752, 2017.

[2] L. Boominathan, S. S. Kruthiventi, and R. V. Babu. Crowdnet: A deep convolutional network for dense crowd counting. In *ACM international conference on Multimedia*, pages 640–644, 2016.

[3] X. Cao, Z. Wang, Y. Zhao, and F. Su. Scale aggregation network for accurate and efficient crowd counting. In *European Conference on Computer Vision*, pages 734–750, 2018.

[4] K. Chen, C. C. Loy, S. Gong, and T. Xiang. Feature mining for localised crowd counting. In *British Machine Vision Conference*, volume 1, page 3, 2012.

[5] M. Enzweiler and D. M. Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2179–2195, 2008.

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

[7] J. Gao, Q. Wang, and X. Li. Pcc net: Perspective crowd counting via spatial convolutional network. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(10):3486–3498, 2019.

[8] D. Guo, K. Li, Z.-J. Zha, and M. Wang. Dadnet: Dilated-attention-deformable convnet for crowd counting. In *ACM International Conference on Multimedia*, pages 1823–1832, 2019.

[9] H. Idrees, I. Saleemi, C. Seibert, and M. Shah. Multi-source multi-scale counting in extremely dense crowd images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2547–2554, 2013.

[10] 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. In *European Conference on Computer Vision*, pages 532–546, 2018.

[11] X. Jiang, Z. Xiao, B. Zhang, X. Zhen, X. Cao, D. Doermann, and L. Shao. Crowd counting and density estimation by trellis encoder-decoder networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6133–6142, 2019.

[12] X. Jiang, L. Zhang, M. Xu, T. Zhang, P. Lv, B. Zhou, X. Yang, and Y. Pang. Attention scaling for crowd counting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4706–4715, 2020.

[13] V. Lempitsky and A. Zisserman. Learning to count objects in images. *Advances in Neural Information Processing Systems*, 23:1324–1332, 2010.

[14] Y. Li, X. Zhang, and D. Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1091–1100, 2018.

[15] J. Liu, C. Gao, D. Meng, and A. G. Hauptmann. Decidenet: Counting varying density crowds through attention guided detection and density estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2018.

[16] L. Liu, H. Wang, G. Li, W. Ouyang, and L. Lin. Crowd counting using deep recurrent spatial-aware network. *arXiv preprint arXiv:1807.00601*, 2018.

[17] W. Liu, M. Salzmann, and P. Fua. Context-aware crowd counting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5099–5108, 2019.

[18] L. Rong and C. Li. Coarse and fine-grained attention network with background-aware loss for crowd density map estimation. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3675–3684, 2021.

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

[20] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883, 2016.

[21] Z. Shi, P. Mettes, and C. G. Snoek. Counting with focus for free. In *IEEE/CVF International Conference on Computer Vision*, pages 4200–4209, 2019.

[22] P. Siva, M. Javad Shafiee, M. Jamieson, and A. Wong. Real-time, embedded scene invariant crowd counting using scale-normalized histogram of moving gradients (HoMG). In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 67–74, 2016.

[23] V. B. Subburaman, A. Descamps, and C. Carincotte. Counting people in the crowd using a generic head detector. In *IEEE International Conference on Advanced Video and Signal-Based Surveillance*, pages 470–475, 2012.

[24] M. Tan and Q. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *PMLR International Conference on Machine Learning*, pages 6105–6114, 2019.

[25] P. Thanasutives, K.-i. Fukui, M. Numao, and B. Kijsirikul. Encoder-Decoder Based Convolutional Neural Networks with Multi-Scale-Aware Modules for Crowd Counting. In *IEEE International Conference on Pattern Recognition*, pages 2382–2389, 2021.

[26] R. R. Varior, B. Shuai, J. Tighe, and D. Modolo. Multi-scale attention network for crowd counting. *arXiv preprint arXiv:1901.06026*, 2019.

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

[28] B. Wang, H. Liu, D. Samaras, and M. Hoai. Distribution matching for crowd counting. *arXiv preprint arXiv:2009.13077*, 2020.

[29] C. Wang, H. Zhang, L. Yang, S. Liu, and X. Cao. Deep people counting in extremely dense crowds. In *ACM International Conference on Multimedia*, pages 1299–1302, 2015.

[30] Q. Wang and T. P. Breckon. Crowd Counting via Segmentation Guided Attention Networks and Curriculum Loss. *arXiv preprint arXiv:1911.07990*, 2019.

[31] L. Wen, D. Du, P. Zhu, Q. Hu, Q. Wang, L. Bo, and S. Lyu. Detection, Tracking, and Counting Meets Drones in Crowds: A Benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[32] Y. Zhang, C. Zhou, F. Chang, A. C. Kot, and W. Zhang. Attention to head locations for crowd counting. In *International Conference on Image and Graphics*, pages 727–737. Springer, 2019.

[33] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma. Single-image crowd counting via multi-column convolutional neural network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 589–597, 2016.

[34] T. Zhao and R. Nevatia. Bayesian human segmentation in crowded situations. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages II–459, 2003.