CHAPTER 3
VIDEO-BASED CROWD COUNTING AND DENSITY ESTIMATION USING DEEP LEARNING TECHNIQUES

3.1 Introduction

In this chapter, two video-based CCDE methods based on deep learning have been proposed. The first methodology, entitled "AMS-CNN: Attentive Multi-Stream CNN for Video-based Crowd Counting," is designed to exploit spatial, temporal, and spatial foreground features from the video sequences. All these multi-features are fused to enhance the quality of feature modeling for crowd counting. The proposed model minimizes the effect of cluttered background by extracting spatial foreground features. In addition, the attention mechanism is designed to pay attention to each feature type's response to crowd counting and improves the model's performance.

The second method, "A Novel Cascaded Deep Architecture with Weak-Supervision for Video Crowd Counting," is proposed. The proposed crowd counting model consists of cascading two deep models trained one after another. This model can handle crowd shape change due to perspective distortion, minimize the effect of the cluttered background, and consider both the local and global crowd distribution for crowd counting. The model also utilizes a Weak-Supervision learning mechanism to minimize the error caused due to the point-level annotation during ground-truth density map generation. The proposed models are evaluated on three publicly available benchmark video-based crowd counting datasets: the Mall [57], the Venice [4], and the UCSD [59]. In addition, an extensive ablation study is performed to show the effectiveness of the proposed models.

The detailed explanations of the two proposed methods are illustrated in the subsequent subsections.

3.2 AMS-CNN: Attentive Multi-Stream CNN for Video-based Crowd Counting

The spatial and temporal features are essential for video-based CCDE [20, 94, 95] but will not improve the model's accuracy because the crowd counting system is mainly affected by the cluttered background. So, in addition to spatial-temporal features, features corresponding to the foregrounds of the crowd scenes must be considered. Instead of adopting a simple multi-cue feature fusion strategy without understanding each feature type's response for the CCDE, an attention-based density map fusion strategy should be adopted for the CCDE to improve the performance of the crowd counting system. These are the main motivations behind the design of AMS-CNN.

3.2.1 Proposed Method and Model

The proposed AMS-CNN constitutes four modules:

- Multi-Stream deep CNN (MS-CNN).
- Stream-wise Attentive Density Map Module (SADMM).
- Relative Average Attentive Density (RAAD) map layer.
- Final Density Map Generation Module (FDMGM).

The architecture of the proposed model is illustrated in Figure 3.1. The detailed structure of SADMM is given in Figure 3.2. The main motive for designing the MS-CNN is to extract the deep spatial, temporal, and spatial-foreground features from three cues of the video dataset: frames, the volume of frames, and the foregrounds of the frames. The MS-CNN has three streams: Spatial Stream, Temporal Stream, and Spatial Foreground Map (FM) stream (see Figure 3.1). The frame, the frame's volume, and the frame's

foreground image at timestamp t are inputted individually to Spatial-Stream, Temporal-Stream, and Spatial Foreground Stream.

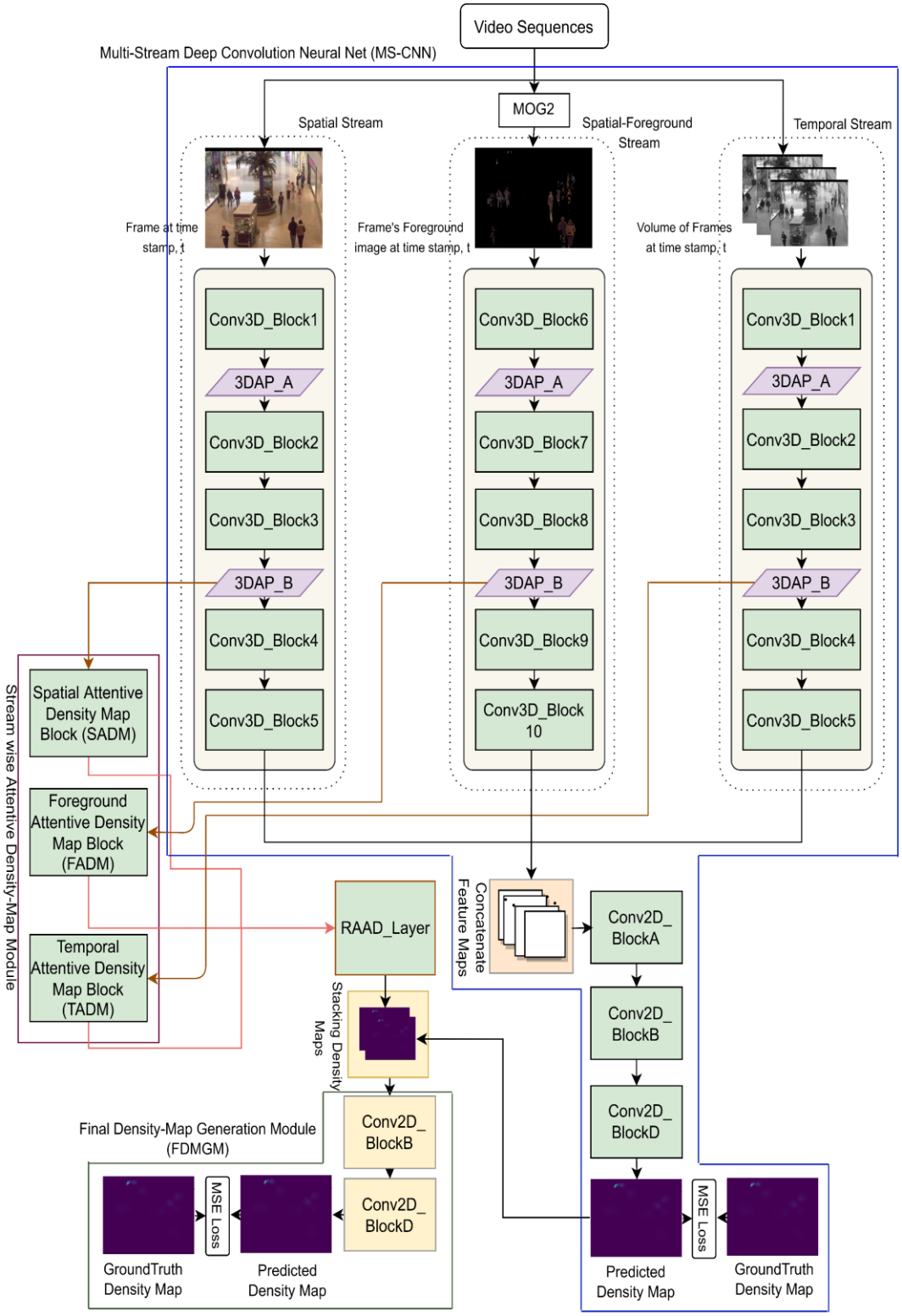

Figure 3.1: Architecture of the proposed AMS-CNN model

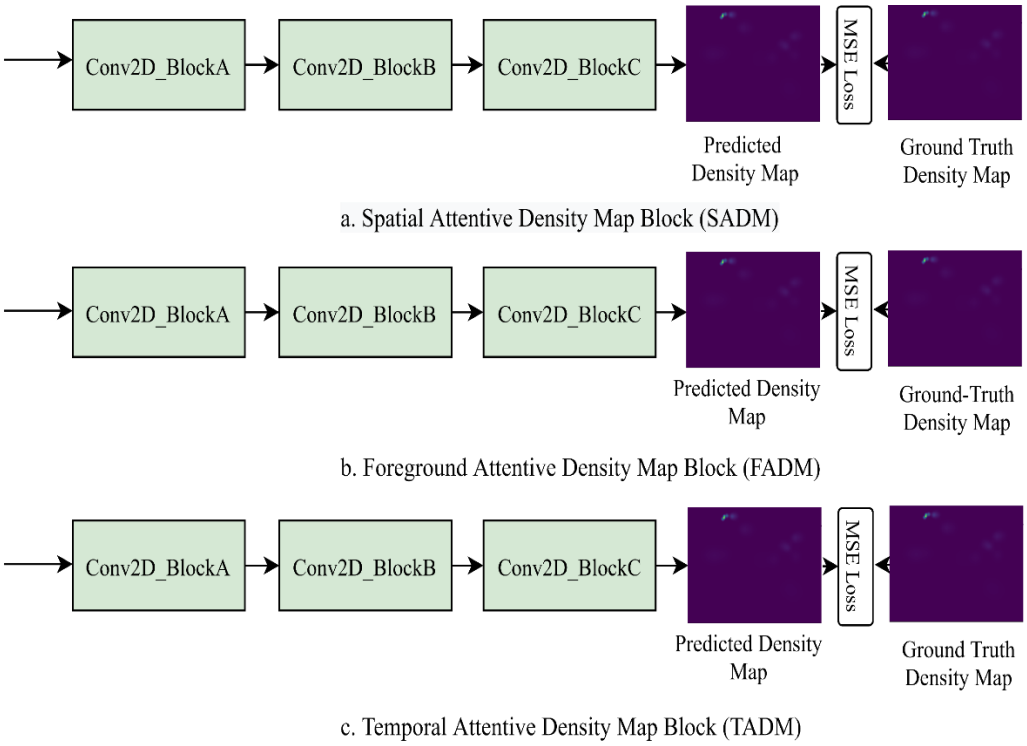

Figure 3.2: Blocks of proposed AMS-CNN

The Spatial-Stream, Temporal-Stream, and Spatial-Foreground-Stream are designed using 3D-CNN. The features of the three streams are concatenated and followed by three stages of 2D convolution (Conv2D) blocks to generate crowd density maps. Further, to improve the accuracy of the proposed model, a stream-wise attentive density map module (SADMM) is proposed. The SADMM learns three stream-wise attentive density maps using three stream-wise attention blocks: the spatial attentive density map (SADM) block, temporal attentive density map (TADM) block, and foreground attentive density map (FADM) block. Each of the three attention blocks contains three stages of 2D-CNN blocks and is intended to generate crowd density maps.

The three stream-wise attentive density maps are fused by obtaining their relative average using a learnable relative averaged attentive density-map (RAAD) layer. Finally, the relative averaged attentive density map is stacked with the density map from the MS-CNN and inputted into the final density map generation module FDMGM). The FDMGM

contains two stages of 2D-CNN blocks, where its second block provides final density maps. The proposed architecture is trained in an end-to-end manner.

The section is further divided into the following subsections to illustrate the working of the AMS-CNN.

- Detail Architecture of AMS-CNN.
- Pre-processing.
- Loss Function for MS-CNN.
- Loss Functions for SADM, FADM, and TADM
- Final Loss Function.
- Training or Learning Process.
- Handling Overfitting.

3.2.1.1 Detail Architecture of AMS-CNN

Each of the three streams of the MS-CNN contains five Conv3D blocks. Each of the Conv3D blocks has three layers: a Conv3D layer, an activation (ReLU) layer, and a batch normalization (BN) layer. Each stream has two 3D average pooling layers (3DAP-A and 3DAP-B). The 3DAP-A and 3DAP-B downscale their input features to $\frac{1}{4}$ and $\frac{1}{8}$ of their shapes, respectively. All the layer's detail is mentioned in Table 3.1. The 3DAP-A and 3DAP-B are followed after Conv3D_Block1 and Conv3D_Block3, respectively.

The padding property of all the layers has been set as "same." The features of the three streams are concatenated and inputted into a three-stage CNN module which contains cascading of Conv2D_BlockA, Conv2D_BlockB, and Conv2D_BlockD. The Conv2D_BlockC is used for crowd density estimation for the MS-CNN. Each

Conv2D_BlockA and Conv2D_BlockB has two layers: Conv2D followed by Rectified Linear Unit (ReLU) activation layer. The Conv2D_BlockD has one kernel with a size (1×1) and is used to generate a crowd density map.

Table 3.1: Layer Details of AMS-CNN

Block Names	**Layers Within the Block**	**Number of Kernels**	**Size of Kernel**	**Block Names**	**Layers Within the Block**	**Number of Kernels**	**Size of Kernel**
Conv3D_Block1	Conv3D	40	(5, 5, 5)	Conv3D_Block9	Conv3D	70	(3, 3, 3)
	ReLU	NA			ReLU	NA	
	BN				BN		
Conv3D_Block2	Conv3D	50	(4, 4, 4)	Conv3D_Block10	Conv3D	80	(3, 3, 3)
	ReLU	NA			ReLU	NA	
	BN				BN		
Conv3D_Block3	Conv3D	60	(4, 4, 4)	Conv2D_BlockA	Conv2D	20	(3, 3)
	ReLU	NA			ReLU	NA	
	BN						
Conv3D_Block4	Conv3D	70	(3, 3, 3)	Conv2D_BlockB	Conv2D	40	(3, 3)
	ReLU	NA			ReLU	NA	
	BN						
Conv3D_Block5	Conv3D	80	(3, 3, 3)	Conv2D_BlockC	Conv2D	1	(1, 1)
	ReLU	NA			Sigmoid	NA	
	BN						
Conv3D_Block6	Conv3D	40	(3, 3, 3)	Conv2D_BlockD	Conv2D	1	(1, 1)
	ReLU	NA			Sigmoid	NA	
	BN						
Conv3D_Block7	Conv3D	50	(3, 3, 3)	3DAP_A	3D Average Pooling	NA	(2, 2, 2)
	ReLU	NA					
	BN						
Conv3D_Block8	Conv3D	60	(3, 3, 3)	3DAP_B	3D Average Pooling	NA	(4, 4, 4)
	ReLU	NA					
	BN						

To improve the accuracy of the proposed model, we introduced a SADMM module that contains three blocks: SADM, FADM, and TADM. Each of these blocks contains a cascading of three Conv2D blocks: Conv2D_BlockA, Conv2D_B, and Conv2D_BlockC. The Conv2D_BlockC has two layers: a Conv2D followed by a Sigmoid activation. We have chosen the Sigmoid activation instead of ReLU for generating sigmoidal stream-wise attentive density-maps. Then the RAAD layer takes all the three-stream-wise attentive density maps and generates relative average attentive

density maps. Finally, the relative averaged attentive density map is stacked with the density map obtained by the MS-CNN followed and fed into the FDMGM module. The FDMGM module has two Conv2D blocks: Conv2D_BlockB and Conv2D_BlockD. The Conv2D_BlockD has only one Conv2D layer with only one kernel of size (1×1) and generates the final crowd density map. We have mentioned the details of the layers in Table 3.1.

3.2.1.2 Pre-processing

The datasets have been pre-processed before training the AMS-CNN. We need three different video cues to input on three streams,

- The RGB color frame for the Spatial-Stream
- The volume of frames for the Temporal-Stream and
- The foreground image of the frame at the timestamp, t for the Spatial (FM)-Stream.

Let the video dataset contains a T number of RGB frames. We resized the RGB frame into size [200×200×3]. Let the resized frames are represented by a set $RF = \{rf_1, rf_2, \dots, rf_T\}$. Next, we obtained the frames' foreground image by applying the Mixture of Gaussian-2 (MOG2) [168] using OpenCV. The process of obtaining the foreground image is illustrated in Algorithm 3.1. Here, an OpenCV function (cv2.createBackgroundSubtractorMOG2) is used to create the background model for the video dataset. Then for each frame, we used another method of OpenCV, i.e., apply() to obtain the frame's foreground mask. Each frame's foreground image is obtained by performing elementwise multiplication of the binary foreground mask with every channel of the resized frame.

Algorithm-3.1 Obtaining foreground images from the frames

Input: Resized frameset, $RF = \{rf_1, rf_2, \dots, rf_T\}$, where T is the total number of frames

Output: A set of foreground images from RF, $FG = \{fg_1, fg_2, \dots, fg_T\}$.

Parameters: history, variance threshold ($varThreshold$), detect shadows ($detectShadows$)

MOG2=cv2.createBackgroundSubtractorMOG2($history = 5$, $varThreshold = 128$,$detectShadows = True$) [168]

for $i = 1$ *to* T **do**

1. mask = MOG2.apply(rf_i) // To obtain foreground mask
2. r = $rf_i[:,:,1]$ //Obtaining the Red channel of the resized frame
3. g = $rf_i[:,:,2]$ //Obtaining the Green channel of the resized frame
4. b = $rf_i[:,:,3]$ //Obtaining the Blue channel of the resized frame
5. Create a variable fg_i whose size is same as rf_i
6. $fg_i[:,:,1]$ = r .* mask //Here .* means elementwise multiplication
7. $fg_i[:,:,2]$ = g .* mask
8. $fg_i[:,:,3]$ = b .* mask

end for

Let the foreground images of RF are represented by a set $FG = \{fg_1, fg_2, \dots, fg_T\}$. For temporal feature extraction, the volume of frames (grayscale) is obtained at the timestamp, t, which is composed of frames at a time, $t, t-1, \dots$. The size of the volume of three, four, and five frames are considered for the experiment. Let the volume of frames is represented by the set $VF = \{vf_1, vf_2, \dots, vf_T\}$. Now the three cues of the video dataset (RF, VF, and FG) are inputted into the Spatial stream, the Temporal stream, and the Spatial-Foreground stream, respectively.

3.2.1.3 Loss Function for MS-CNN

Let θ_{MS-CNN} represent all the learnable parameters corresponding to the MS-CNN. For the given three sets of image cues: RF, VF, and FG, let the predicted density map of

the MS-CNN is represented as, $F_i^{MS-CNN}(rf_i, vf_i, fg_i; \theta_{MS-CNN})|_{i=1,2,\ldots,T}$, where T is the total number of frames in the video dataset. Now we need to find the loss between the predicted density map (F^{MS-CNN}) and ground-truth density map (GT) for the proposed MS-CNN. The MSE is used to capture the loss between the predicted and the ground-truth density maps. Let the loss function is represented by using Equation 3.1.

$$Loss_1 = Loss(\theta_{MS-CNN}) = \frac{1}{T}\sum_{i=1}^{T}\left(F_i^{MS-CNN} - gt_i\right)^2 \quad (3.1)$$

3.2.1.4 Loss Functions for SADM, FADM and TADM

The three-stream-wise attention density map modules of SADMM are intended to provide three density maps. Let θ_{SADM}, θ_{FADM}, and θ_{TADM} represent all the learnable parameters of the SADM, FADM, and TADM. Let the feature maps obtained from 3DAP_B layer of Spatial-Stream, Spatial-Foreground Stream, and Temporal-Stream are represented as $fm_{i,3DAP\_B}^{SADM}$, $fm_{i,3DAP\_B}^{FADM}$ and $fm_{i,3DAP\_B}^{TADM}$ respectively, where $i = 1,2,\ldots,T$. Now, for the given feature maps: $fm_{i,3DAP\_B}^{SADM}$, $fm_{i,3DAP\_B}^{FADM}$ and $fm_{i,3DAP\_B}^{TADM}$, let the predicted density maps of the SADM, FADM, and TADM are represented as $F_i^{SADM}\left(fm_{i,3DAP\_B}^{SADM}; \theta_{SADM}\right)$, $F_i^{FADM}\left(fm_{i,3DAP\_B}^{FADM}; \theta_{FADM}\right)$ and $F_i^{TADM}\left(fm_{i,3DAP\_B}^{TADM}; \theta_{TADM}\right)$ respectively, where $i = 1,2,\ldots,T$. The MSE is now used to capture the loss between the predicted density map and the ground-truth density map for the SADM, FADM, and TADM, represented in Equation 3.2, 3.3, and 3.4 respectively.

$$Loss_2 = Loss(\theta_{SADM}) = \frac{1}{T}\sum_{i=1}^{T}\left(F_i^{SADM} - gt_i\right)^2 \quad (3.2)$$

$$Loss_3 = Loss(\theta_{FADM}) = \frac{1}{T}\sum_{i=1}^{T}\left(F_i^{FADM} - gt_i\right)^2 \quad (3.3)$$

$$Loss_4 = Loss(\theta_{TADM}) = \frac{1}{T}\sum_{i=1}^{T}\left(F_i^{TADM} - gt_i\right)^2 \quad (3.4)$$

The RAAD is a learnable module that finds the relative density maps from the three density maps obtained in the SADMM. The size of these three density maps is the

same; let it be $[m \times n]$. The relative average attentive density maps (F_i^{RAAD}) can be obtained using Equations 3.5 to 3.8.

$$w_1 = Sum\left(F_i^{SADM}\right) \tag{3.5}$$

$$w_2 = Sum\left(F_i^{FADM}\right) \tag{3.6}$$

$$w_3 = Sum\left(F_i^{TADM}\right) \tag{3.7}$$

$$F_i^{RAAD} = \left[\frac{w_1}{w_1+w_1+w_1}\right] \times F_i^{SADM} + \left[\frac{w_2}{w_1+w_1+w_1}\right] \times F_i^{FADM} + \left[\frac{w_3}{w_1+w_1+w_1}\right] \times F_i^{TADM} \tag{3.8}$$

Here the symbol $Sum(.)$ is used to find out the sum of all elements of the feature map.

3.2.1.5 Final Loss Function and Optimization

The FDMGM is proposed to obtain the final crowd density map. The two crowd density maps: F_i^{RAAD} and F_i^{MS-CNN}, are concatenated and inputted into the FDMGM. Let θ_{FDMGM} represent all the learnable parameters of the FDMGM. Let the final crowd density map obtained from FDMGM be represented as $F_i^{Final}\left(Concate\left(F_i^{RAAD}, F_i^{MS-CNN}\right); \theta_{FDMGM}\right)$. Here the terminology $Concate(.)$ represents the simple concatenation operation. Next, the MSE loss between the predicted density map and the ground-truth density map for the FDMGM is obtained using Equation 3.9.

$$Loss_5 = Loss(\theta_{FDMGM}) = \frac{1}{T}\sum_{i=1}^{T}\left(F_i^{Final} - gt_i\right)^2 \tag{3.9}$$

Now, the final loss function is obtained by summing all the losses obtained from Equation 9 to 13 and let it is represented as,

$$L_{Final} = Loss_1 + Loss_2 + Loss_3 + Loss_4 + Loss_5 \tag{3.10}$$

Now, the problem can be deduced to an optimization problem where we have to minimize Equation 3.11, which can be represented as.

$$\underset{\theta_{NET}}{argmin}\left[L_{Final}\right] \tag{3.11}$$

Here $\theta_{NET} = [\theta_{MS-CNN}, \theta_{SADM}, \theta_{TADM}, \theta_{FADM}, \theta_{FDMGM}]$ contains all the trainable parameters.

Now Equation 3.11 is minimized using the backpropagation algorithm [169] and Adaptive Momentum (Adam) [170] optimizer. The training is done in a mini-batch manner. We used early-stopping to halt the network and also to avoid overfitting. In early stopping, we have to set a patience parameter. If the monitoring metric, i.e., training loss in our case, is not improved (minimized) even after the patience value, the training will be halted; otherwise, it will be continued. The patience parameter is set to 30.

Overfitting becomes a major concern in geting a robust deep model. In the proposed work, we have used the L_2 norm and early-stopping to overcome the overfitting. The L_2 norm will penalize the weight to regularize the kernel parameters. All the convolution layers of the AMS-CNN have the kernel-regularize as L_2 norm, whose regularize parameter is set to 0.01. Early-stopping is used to halt the training and avoid overfitting.

3.2.2 Experimental Setup

The proposed AMS-CNN has been coded in the PyCharm editor using Keras and TensorFlow. The codes were executed using different computing nodes of the Super Computer, i.e., the Param-Shivay, which is available in the institute. The batch size for the Mall [57], the Venice, and the UCSD was set to 64, 8, and 64, respectively. The learning rate η, momentum of batch normalization, regularized parameter of L_2, and decay rates for first (β_1) and second moment (β_2) of Adam optimizer [170] are initialized to 0.001, 0.95, 0.01, 0.9, and 0.999, respectively. The maximum iteration was 2000. Early stopping with patience of 30 is used to stop the model's training and avoid overtraining the model.

3.2.3 Results Analysis and Discussion

3.2.3.1 The Mall Dataset

The proposed model has been experimented with different sizes of frames' volume (used in the Temporal-Stream). Different sizes of frames' volume (FV) like 3, 4, and 5 are considered. The comparative analysis of results on the Mall dataset [57] is illustrated in the following Table 3.2, where the values in bold letters are the highest in the table. The results of the AMS-CNN are compared with the other 14 state-of-the-art approaches. It can be observed from Table 3.2 that the AMS-CNN with $FV = 5$ achieves better results as compared with AMS-CNN with $FV = 3$ and AMS-CNN with $FV = 4$. The proposed AMS-CNN with $FV = 5$ achieves MAE and RMSE of 2.47 and 3.08, respectively. Figure 3.3 shows a bar graph representing the predicted crowd count on the Mall dataset [57] by the AMS-CNN ($FV = 5$).

Table 3.2: Comparative Analysis of Results on the Mall Dataset [57]

Model Names	MAE	RMSE
LBP + Ridge Regression [57]	6.73	19.18
Kernel Ridge Regression (KRR) [58]	6.61	18.85
Gaussian Process Regression (GPR) [59]	7.15	21.34
Count Forest [70]	5.75	10.88
CNN-MRF [171]	4.66	9.01
Faster R-CNN [172]	4.65	7.26
MCNN [27]	4.74	8.64
CCNN [28]	5.36	9.34
ConvLSTM-nt [12]	2.53	11.
ConvLSTM [12]	2.24	8.5
Bidirectional ConvLSTM [12]	**2.10**	7.6
DAL-SVR [173]	2.40	9.57
DIGCrowd [174]	3.21	16.4
ST-CNN [32]	4.03	5.87
Proposed AMS-CNN ($FV = 3$)	2.94	3.63
Proposed AMS-CNN ($FV = 4$)	2.60	3.25
Proposed AMS-CNN ($FV = 5$)	2.47	**3.08**

It can be noticed from Table 3.2 that the conventional approaches such as LBP + Ridge [57], KRR [58], GPR [59], and Count Forest [70] perform poorly as compared with the deep learning approaches. Among the single image-based techniques like CNN-MRF

[171], Faster R-CNN [172], MCNN [27], CCNN [28], DAL-SVR [173], and DIGCrowd [174], the DAL-SVR [173] performs better with MAE=2.4 and RMSE=9.57. However, the difference between the MAE and RMSE is very high. This is because of the high difference between the ground-truth counts and the DAL-SVR's [173] predicted count.

Among the video-based CCDE approaches such as ConvLSTM-nt [12], ConvLSTM [12], Bidirectional ConvLSTM [12], ST-CNN [32], and the proposed AMS-CNN, the Bidirectional ConvLSTM [12] has the better MAE (=2.10) but very poor RMSE (=7.6). The high RMSE indicated that the Bidirectional ConvLSTM [12] could not predict consistent count and resulted in a high deviation from the ground truth. From Table 3.2, we can find that the proposed AMS-CNN (FV=5) achieves the highest RMSE of 3.08 compared to all the approaches. It places third and fourth highest in MAE compared to video-based CCDE and all in the list, respectively. Nevertheless, the difference between the MAE value of AMS-CNN (FV=5) with Bidirectional ConvLSTM [12], ConvLSTM [12], and DAL-SVR [173] is very less. The AMS-CNN (FV=5) performs far better than the recent video-based CCDE model, i.e., ST-CNN [32]. A significant thing can be observed from Table 3.2 that the difference between ground-truth and predicted count of AMS-CNN (FV=5) is very less, and thus concluded that the proposed model results in consistent prediction and do not result in high deviation from ground-truth as far as the mentioned methods in Table 3.2. Hence, it can be conclude that the proposed model possesses better MAE and RMSE. So, the AMS-CNN (FV=5) model provides a comparatively better solution.

3.2.3.2 The Venice Dataset

Table 3.3 shows a comparative analysis of several approaches on the Venice dataset [4]. The values in bold letters in Table 3.3 are the highest in the table. The

proposed AMS-CNN with FV=5 gets better results compared to AMS-CNN with FV=3 and AMS-CNN with FV=4.

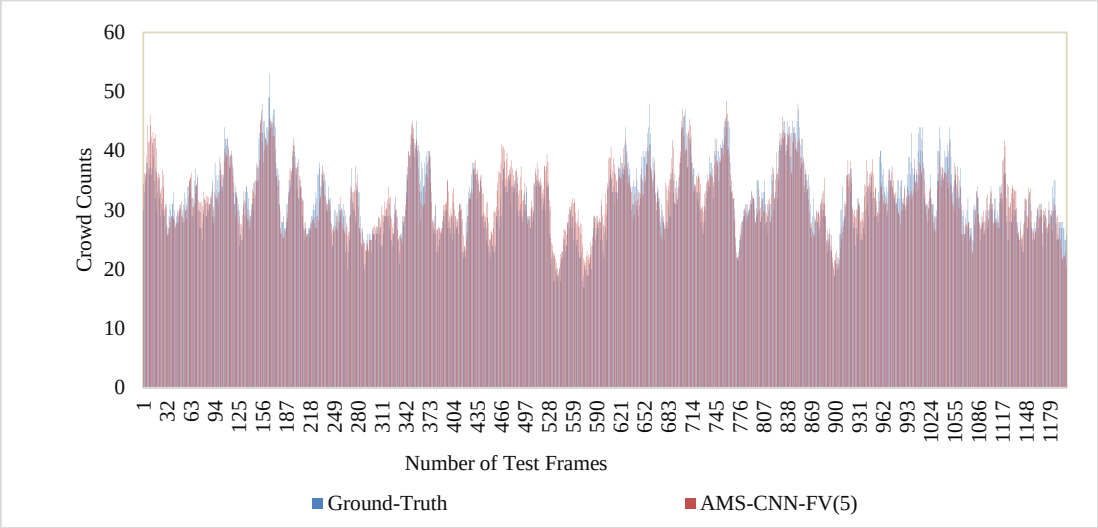

Figure 3.3: Predicted crowd counts on the Mall dataset [57].

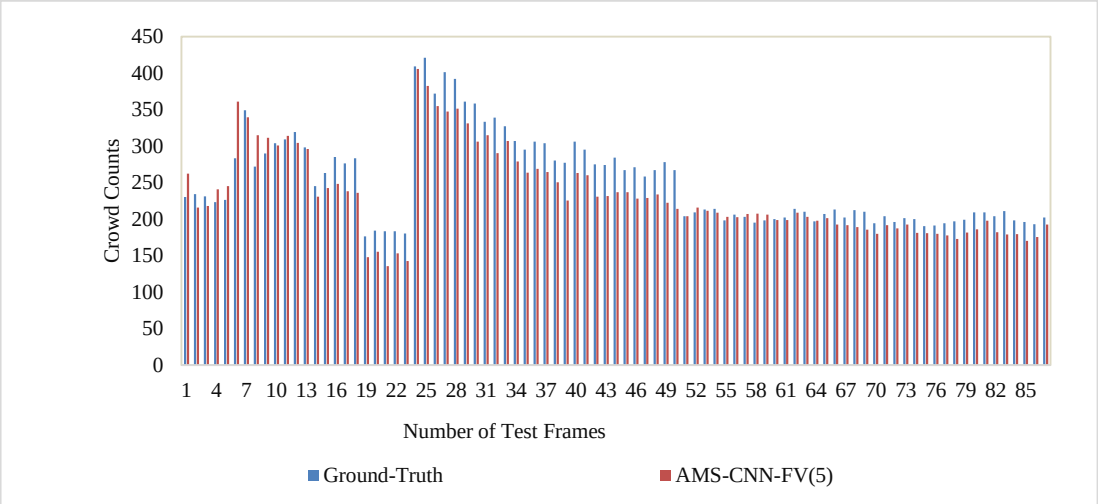

Figure 3.4: Predicted crowd counts on the Venice dataset [4]

Table 3.3: Comparative Analysis of Results on the Venice Dataset [4].

Model Name	**MAE**	**RMSE**
MCNN [27]	145.4	147.3
Switch-CNN [85]	52.80	59.50
CSR-Net [94]	35.80	50.00
ECAN [4]	**20.50**	29.90
Proposed AMS-CNN ($FV = 3$)	30.52	42.58
Proposed AMS-CNN ($FV = 4$)	28.79	37.66
Proposed AMS-CNN ($FV = 5$)	23.64	**28.75**

Among all models mentioned in Table 3.3, the ECAN [4] achieves the highest MAE of 20.5, and the AMS-CNN (FV=5) achieves the highest RMSE of 28.75. Figure 3.4 shows the bar graph representation of predicted count against ground-truth count on the Venice dataset [4].

The proposed model performs better than the state-of-the-art approaches like MCNN [27] and Switch-CNN [85], and CSR-Net [94]. Although ECAN [4] obtains the highest MAE, the difference between MAE and RMSE is more than the proposed model. This is because the ECAN [4] could not make a consistent prediction and generally results in a very high deviation of prediction count from the ground-truth count. So, by considering these facts, the proposed AMS-CNN (FV=5) performs better than the state-of-the-art models [4][27][85][94] in the presence of very high-density crowd scenes.

3.2.3.3 The UCSD Dataset

The experimental results obtained by the proposed model are compared with 16 state-of-the-art techniques and illustrated in Table 3.4. The values mentioned in bold letters in Table 3.4 are the best in the table. The proposed AMS-CNN with FV=5 performs better than AMS-CNN with FV=3 and FV=4. The AMS-CNN (FV=5) obtains MAE and RMSE of 1.46 and 1.82, respectively. Figure 3.5 shows a bar graph that presents the ground truth count versus predicted count for the UCSD dataset [168]. According to Table 3.4, we can find that the deep learning-based approaches [12], [28], [85], [90], [111], [163], [173], [175]–[177] perform better than the traditional approaches [57], [59], [60], [68], [70].

As far as deep learning-based CCDE techniques are concerned, we can find that the video-based CCDE techniques [12], [111] perform better than the single-image-based CCDE approaches [28], [85], [90], [173], [175]–[177].

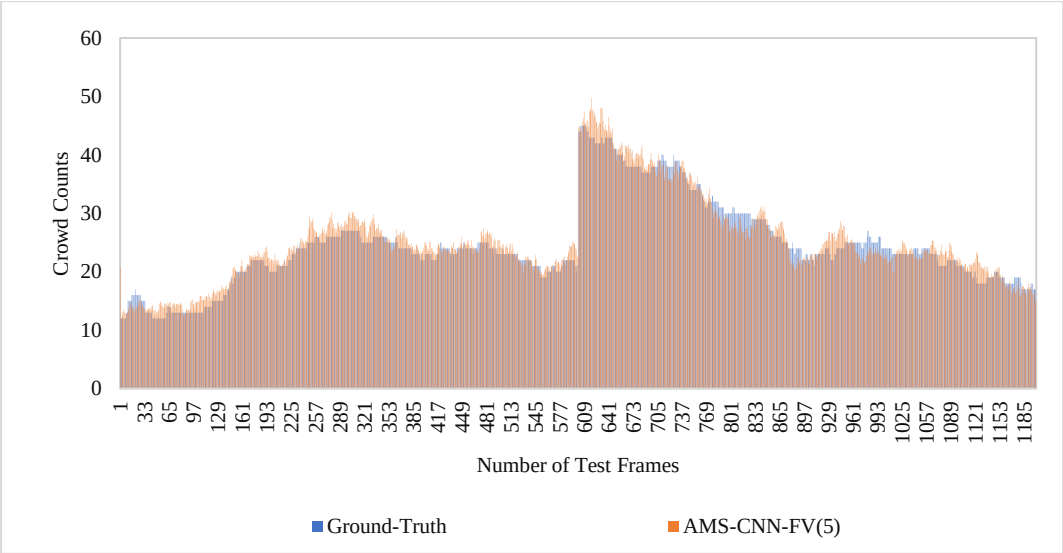

Figure 3.5: Predicted crowd counts on the UCSD dataset [59]

Table 3.4: Comparison of results of several models on the UCSD Dataset [59]

Model Name	**MAE**	**RMSE**
Gaussian process regression [59]	2.24	7.97
Ridge regression [57]	2.25	7.82
Cumulative attribute regression (CAR) [60]	2.07	6.90
Density map + MESA [68]	1.70	-
Count forest [70]	1.60	4.40
CCNN [28]	1.5125	-
CrossCrowdNet [175]	1.60	3.31
DAL-SVR [173]	1.29	2.10
Switch-CNN [85]	1.62	2.1
ConvLSTM-nt [12]	1.73	3.52
ConvLSTM [12]	1.30	1.79
Bidirectional ConvLSTM [12]	**1.13**	**1.43**
AFP [177]	1.16	2.29
FCN-rLSTM [111]	1.54	3.02
MDMF-CC [90]	1.35	1.88
MS-GAN [176]	1.78	3.03
Proposed AMS-CNN ($FV = 3$)	2.46	3.32
Proposed AMS-CNN ($FV = 4$)	1.88	2.57
Proposed AMS-CNN ($FV = 5$)	1.46	1.82

Among the video-based CCDE approaches, the bidirectional ConvLSTM [12] performs better with MAE=1.13 and RMSE=1.43. The proposed model places second as far as video-based CCDE models are concerned. One of the reasons for getting such a result may be due to the fact that the UCSD contains low-resolution crowd scenes with

Grayscale frames. So, we can conclude that the proposed model performs reasonably well compared to other state-of-the-art approaches.

3.2.3.4 Ablation Study

An ablation study has been performed on the proposed AMS-CNN to show the performance of different modules of the proposed model. We developed different models based on different possible combinations of modules of AMS-CNN. We have considered the following possible combinations during the ablation study.

- The multi-stream deep CNN (MS-CNN) (without stream-wise attentive density map module).
- The spatial attentive density map (SADM) guided Spatial-Stream (SADM-Spatial-Stream).
- The foreground attentive density map (FADM) guided Spatial-Foreground-Stream (FADM-Foreground-Map-Stream).
- The temporal attentive density map (TADM) guided Temporal-Stream (TADM-Temporal-Stream).

Table 3.5: Comparison of results of different models during ablation study

The model used for Ablation study	**The Mall Dataset**		**The Venice Dataset**		**The UCSD Dataset**	
	MAE	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
SADM-Spatial-Net	4.64	6.01	52.44	64.65	2.76	3.60
FADM-Spatial Foreground Net	4.73	6.07	48.16	70.77	2.77	4.65
TADM-Temporal Net ($FV = 5$)	2.98	3.77	37.00	41.67	2.82	3.65
MS-CNN ($FV = 5$)	3.58	4.45	35.85	48.77	2.23	2.84
AMS-CNN ($FV = 3$)	2.94	3.63	30.522	42.58	2.46	3.32
AMS-CNN ($FV = 4$)	2.60	3.25	28.79	37.66	1.88	2.57
AMS-CNN ($FV = 5$)	**2.47**	**3.08**	**23.64**	**28.75**	**1.46**	**1.82**

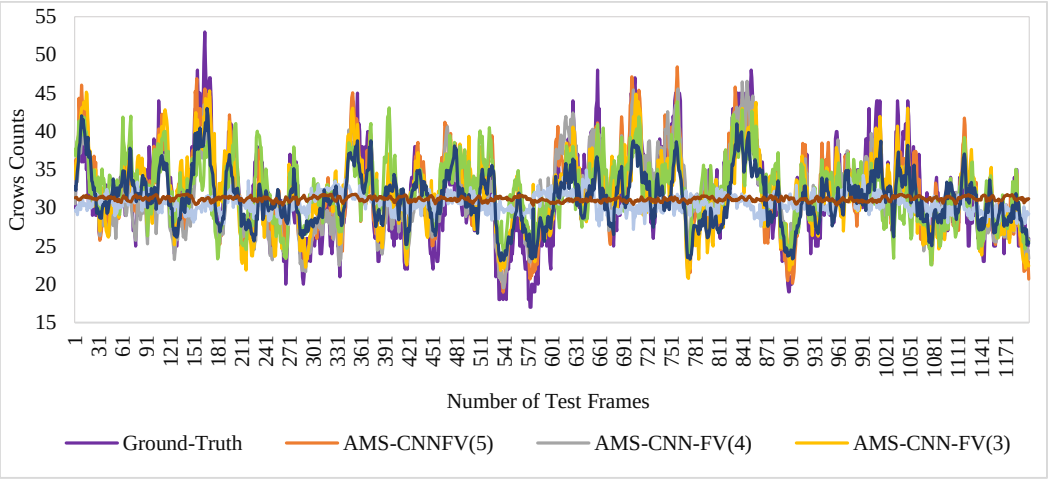

Figure 3.6: Predicted Crowd Counts of different models on the Mall dataset [57]

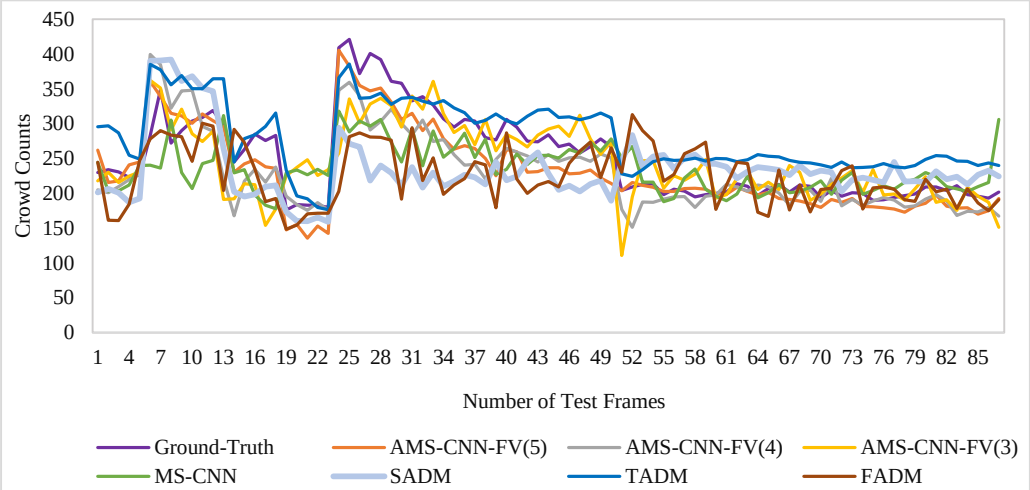

Figure 3.7: Predicted Crowd Counts of different models on the Venice dataset [4]

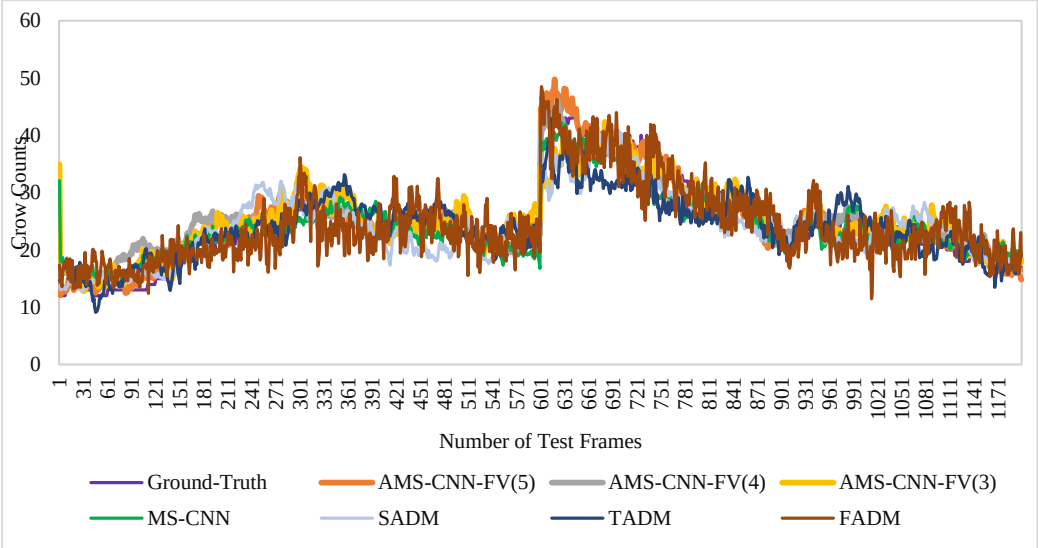

Figure 3.8: Predicted Crowd Counts of different models on the UCSD dataset [59]

Table 3.5 shows a comparison of results of different models during the ablation study on the Mall [57], the Venice [4], and the UCSD [59]. The values mentioned in bold letters in Table 3.5 are the best in the table. Figures 3.6, 3.7 and 3.8 show the graphical representation of predicted crowd counts versus ground-truth counts by different ablation study models on the Mall dataset [57], the Venice Dataset [4], and the UCSD dataset [59], respectively. It can be observed from Table 3.5 that the AMS-CNN (FV=5) performs better among other modules. Individual modules other than AMS-CNN (FV=5) are not capable of providing better results. So, it can be summarized that the AMS-CNN (FV=5) takes all the advantages of its modules, and the counting system's performance is improved.

3.3 A Novel Cascaded Deep Architecture with Weak-Supervision for Video Crowd Counting

The state-of-the-art video-based CCDE approaches [12], [32], [33], [111] based on the DMR technique, extract spatial and temporal features from the crowd videos. However, such approaches are lacking in extracting scale-invariant features to handle scale changes due to perspective distortion, do not take measures to minimize the effect of the cluttered background, and are error-prone due to the manual process for point label annotations for crowd heads. This study proposes a novel cascaded deep architecture with weak supervision for video crowd counting to address these issues. The following subsection discusses the detail of the proposed model.

3.3.1 Proposed Method and Model

This work proposes a novel cascading of two deep models for video-based CCDE. The two deep models are trained one after another. The first deep model, also termed the Local Density map Regression (LDR) model, can handle crowd head shape change,

minimize background influences, and generate crowd density maps using three sub-modules:

- The Mani-Fold Multiscale spatial-temporal Feature Fusion Module (MF-MSFFM)
- The Head Attention Module (HAM), and
- The Density Map Regression Module (DMRM)

The second deep model, i.e., the Global Crowd Count Regression (GCCR) module adopts weakly-supervision strategy, generates the global crowd count values using the proposed MLP module. The overall block diagram of the proposed model is illustrated in Figure 3.9.

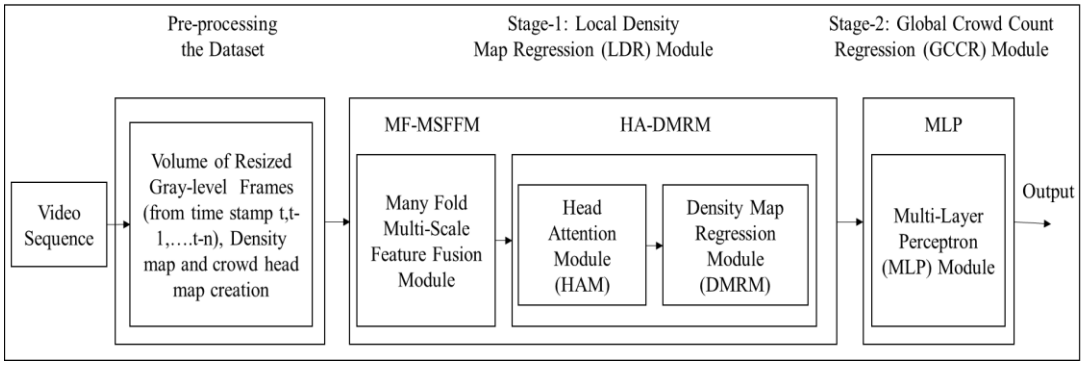

Figure 3.9: Block diagram of the proposed cascaded deep model

According to Figure 3.9, the proposed deep model contains the following main components.

- Pre-processing the dataset.
- Crowd density map generation using the LDR Module
    - Mani-Fold Multi-Scale spatial-temporal Feature Fusion Module (MF-MSFFM)
    - Head Attention guided Density Map Regression Module (HA-DMRM)
        - Head Attention module (HAM)
        - Density Map Regression Module (DMRM)

- Scene-level crowd counting using Weakly-Supervised Global crowd counting regression (GCCR) Module

3.3.1.1 Pre-processing.

At first, the video dataset is converted into frames. The RGB frames are resized to $[200 \times 200]$ and converted into grayscale images. Let the total N number of resized grayscale frames be denoted by $S = \{s_1, s_2, \ldots\ldots, s_N\}$. The candidates for the proposed model are the volume of frames; let these be represented by a set $V = \{v_1, v_2, \ldots\ldots, v_t, \ldots\ldots v_N\}$. Here volume of frames at the timestamp, t is obtained by stacking frames at the timestamp, $t, t-1, \ldots, t-n$, where n is the size of the volume. The experiments are conducted with sizes n of 3, 4, 5, and 6 consecutive frames. Apart from the volume of frames, ground-truth crowd density maps and crowd head maps are also required for the proposed model. The ground-truth crowd density maps are obtained using Equation 2.3. The crowd head maps are obtained by following the work of HA-CCN [89]. The size of the crowd head maps is the same as the density maps. Let a set $HM = \{hm_1, hm_2, \ldots\ldots, hm_N\}$, represent the crowd head maps for the N number of frames.

3.3.1.2 Working of the LDR Module

The detailed architecture of the proposed model is illustrated in Figures 3.10 and 3.11. The main motive behind the LDR model is to obtain crowd density maps by minimizing the background influences.

3.3.1.2.1 Architectural details

According to Figure 3.10, the LDR module constitutes two modules: MF-MSFFM and HA-DMRM. The HA-DMRM has submodules like HAM and DMRM. The MF-MSFFM

is designed with three columns of multi-layer 3D-Atrous-Net with different sizes of receptive fields.

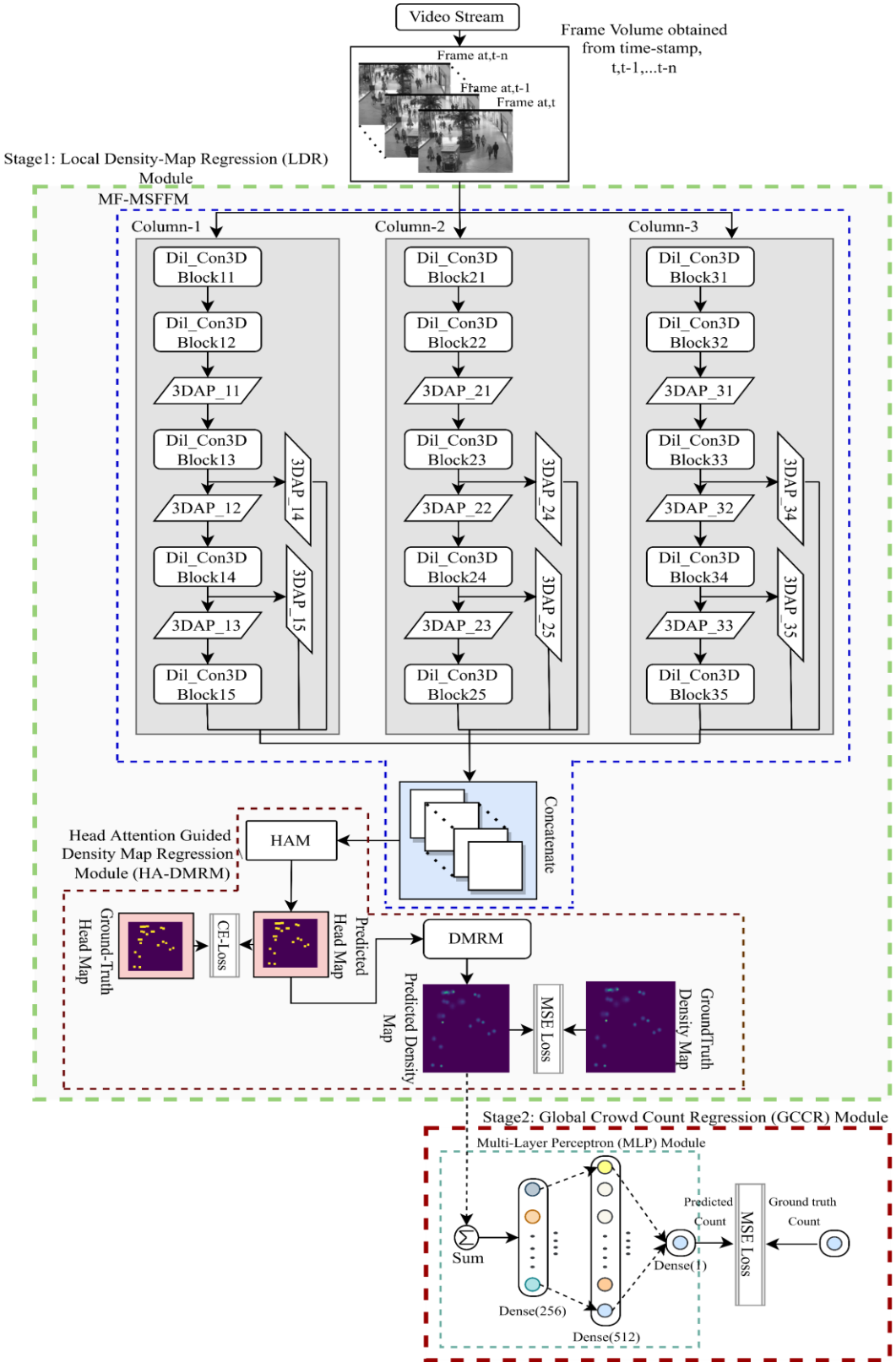

Figure 3.10: Details of layers used in the proposed architecture

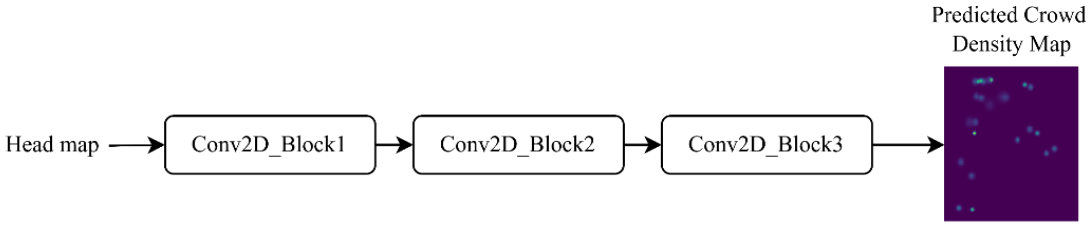

Figure 3.11: Details of DMRM

Table 3.6: Details of layers used in the proposed architecture

Blocks Name	**Blocks Details**	**Number of Kernels**	**Kernel Size**	**Dilation Rate**
Dil_Con3DBlock11	Dil_Con3D	20	3, 3, 3	5, 5, 5
	BN	NA		
	ReLU			
Dil_Con3DBlock12	Dil_Con3D	30	3, 3, 3	5, 5, 5
	BN	NA		
	ReLU			
Dil_Con3DBlock13	Dil_Con3D	60	3, 3, 3	4, 4, 4
	BN	NA		
	ReLU			
Dil_Con3DBlock14	Dil_Con3D	80	3, 3, 3	4, 4, 4
	BN	NA		
	ReLU			
Dil_Con3DBlock15	Dil_Con3D	100	3, 3, 3	4, 4, 4
	BN	NA		
	ReLU			
Dil_Con3DBlock21	Dil_Con3D	20	3, 3, 3	4, 4, 4
	BN	NA		
	ReLU			
Dil_Con3DBlock22	Dil_Con3D	30	3, 3, 3	4, 4, 4
	BN	NA		
	ReLU			
Dil_Con3DBlock35	Dil_Con3D	100	2, 2, 2	1, 1,1
	BN	NA		
	ReLU			
Conv2D_Block1	Conv2D	20	2, 2	
	ReLU	NA		
Dil_Con3DBlock23	Dil_Con3D	60	3, 3, 3	4, 4, 4
	BN	NA		
	ReLU			
Dil_Con3DBlock24	Dil_Con3D	80	3, 3, 3	3, 3, 3
	BN	NA		
	ReLU			
Dil_Con3DBlock25	Dil_Con3D	100	3, 3, 3	3, 3, 3
	BN	NA		
	ReLU			
Dil_Con3DBlock31	Dil_Con3D	20	2, 2, 2	2, 2, 2
	BN	NA		
	ReLU			

Dil_Con3DBlock32	Dil_Con3D	30	2, 2, 2	2, 2, 2
	BN	NA		
	ReLU			
Dil_Con3Dblock33	Dil_Con3D	60	2, 2, 2	2, 2, 2
	BN	NA		
	ReLU			
Dil_Con3DBlock34	Dil_Con3D	80	2, 2, 2	1, 1,1
	BN	NA		
	ReLU			
Conv2D_Block1	Conv2D	20	2, 2	NA
	ReLU	NA		
Conv2D_Block2	Conv2D	30	2, 2	NA
	ReLU	NA		
Conv2D_Block2	Conv2D	1	1, 1	NA
	Sigmoid	NA		

The inputs to the multicolumn Atrous-Net are the volumes of frameset V. Each column contains five dilation 3D convolution blocks, identified as $Dil\_Con3DBlock_{i,j}$ $(here, the\ column\ index\ i = 1,2,3\ and\ block\ index\ j = 1,2,3,4,5)$. Each dilation 3D convolution block constitutes three things: A 3D dilation convolution layer (Dil_Conv3D), a batch normalization (BN) layer, and a ReLU activation layer. The 3D average pooling (3DAP) layers are used to downscale the feature maps. In each column i $(i = 1,2,3)$, the pooling layers i.e., $3DAP_{i,1}$ follows after $Dil\_Con3DBlock_{i,2}$, the $3DAP_{i,2}$ follows after $Dil\_Con3DBlock_{i,3}$ and the $3DAP_{i,3}$ follows after $Dil\_Con3DBlock_{i,4}$. These average pooling layers downscale the feature maps to its half. The skip connections are used to fuse in-layer multiscale features. For this, in each column i $(i = 1,2,3)$, the $3DAP_{i,4}$ follows after $Dil\_Con3DBlock_{i,3}$ and the $3DAP_{i,5}$ follows after $Dil\_Con3DBlock_{i,4}$. The $3DAP_{i,4}$, and $3DAP_{i,5}$ $(for, i = 1,2,3)$ downscale feature maps to $\frac{1}{4}$ and $\frac{1}{2}$ respectively. The layer details are illustrated in Table 1. The features obtained from $Dil\_Con3DBlock_{i,5}$, $3DAP_{i,4}$ and $3DAP_{i,5}$ from each column i $(i = 1,2,3)$ are fused by concatenating them. The fused multiscale features are forwarded to the HAM to learn the head maps. The HAM contains a merging layer using

Conv2D with a single kernel having a size (1×1). The output of HAM is inputted in the DMRM module. The Detail architecture of DMRM is illustrated in Figure 3.12, and layer details are illustrated in Table 3.6. The DMRM module contains three Conv2D blocks. Each of the first two Conv2D blocks contains Conv2D layers followed by the ReLU activation layer. The Conv2D_Block3 merges the feature maps by using a single kernel having a size (1×1). The Sigmoid activation function is used to activate the feature map of Conv2D_Block3. The output of Conv2D_Block3 provides the predicted density map. The second deep model, i.e., the GCCR module, has an MLP module with three dense layers, each having a dense connection with 256, 512, and 1 neuron. The input to the MLP is the sum of predicted crowd density maps. All the layers of the proposed model are padded with zeros.

3.3.1.2.2 Manifold multiscale spatial-temporal feature extraction

The scale variation due to perspective distortion in videos can be handled by extracting multiscale spatial-temporal features from the crowd scene. In the proposed work, the multiscale spatial-temporal features are extracted in a manifold manner. By following the idea of Zhang et al.[30], a multicolumn (three-column in the proposed model) architecture is proposed for multiscale feature extraction. The three columns of multi-layer 3D Atrous-Net with receptive fields of different sizes are designed. The Atrous-Net is also called as dilated CNN. Each column can provide features of different scales. We have used Atrous-Net instead of normal CNN layers due to the following reasons.

- It can occupy a larger receptive field with lesser parameters.
- The resolution of the output image is not lost.

The features from three different columns are fused. In addition to the multiscale features extracted from three columns, we added multiscale features from two different

layers ($Dil\_Con3DBlock_{i,3}\ and\ Dil\_Con3DBlock_{i,4}$) of each column, i and the value of $i = 1,2\ and\ 3$. To maintain the resolution of features obtained from $Dil\_Con3DBlock_{i,3}\ and\ Dil\_Con3DBlock_{i,4}$ same as the multicolumn features, we downscale these feature maps by $\frac{1}{4}$ and $\frac{1}{2}$ respectively. Let, the multiscale features obtained from the volume of frameset V is represented by a set $F = \{f_1, f_2, \dots \dots, f_N\}$.

3.3.1.2.3 Head attention guided density map regression module (HA-DMRM)

The crowd density estimation models generate the crowd density maps by mapping the learned feature sets onto the ground-truth crowd density maps. However, the ground-truth crowd density maps are obtained from annotated head points. In such a scenario, the ground-truth density maps contain only crowd head distributions. Hence, during regression, if the learned multiscale features are mapped onto the head areas before regressing onto the density maps, we can minimize the influence of cluttered backgrounds. Such types of mapping can be seen in the work of HA-CCN [48]. The proposed HA-DMRM module constitutes two modules: the HAM and the DMRM. The principle design of HAM follows the HA-CCN [48], which is discussed below.

a. The Working of HAM

The HAM contains only a single Conv2D layer with Sigmoid activation. The Conv2D layer provides a single activation map and merges all the incoming feature maps using (1×1) convolution. The output of the sigmoidal activation map will be used for predicting the head maps. Let $\emptyset_{HAM}$, represents all the trainable parameters connecting with the HAM. Let $P_{hm} = \{p_{hm_1}, p_{hm_2}, \dots., p_{hm_N}\}$ represent the set of predicted head maps for N number of input samples.

Let hm_i be the ground truth head map of the i^{th} crowd scene. The loss is captured by using binary Cross-Entropy between the ground-truth head map (hm_i) and the

predicted head map (p_{hm_i}) for the i^{th} crowd scene. Let $Loss_i(hm_i, p_{hm_i})$ be the Cross-Entropy loss (CE-Loss) on the i^{th} crowd scene and can be represented as follows.

$$[Loss_{HAM}]^i = [Loss(\emptyset_{HAM})]^i = Loss_i(hm_i, p_{hm_i}) = -\sum_{p=1}^{Size} hm_{i_p} \log p_{hm_{i_p}} \quad (3.12)$$

, here $size$ represents the size of the output feature map of HAM.

b. The Working of the DMRM

The DMRM module generates crowd density maps for i^{th} sample from the predicted head maps, i.e., p_{hm_i}. The main focus is to generate crowd density maps from de-background head maps. Let $\emptyset_{DMRM}$ be the set of learnable parameters connecting with the DMRM. Let the predicted crowd counts for $the\ i^{th}$ crowd scene be represented by p_i. In the experiment the squared error is used to capture the loss between the predicted and ground-truth crowd counts, which can be represented as follows.

$$[Loss_{DMRM}]^i = [Loss(\emptyset_{DMRM})]^i = Loss_i(p_i, gt_i) = (p_i - gt_i)^2 \quad (3.13)$$

3.3.1.2.4 Optimizing the LDR

The LDR module is trained in an end-to-end manner and batch wise stochastic gradient descent approach is adopted. Let the trainable parameters of the LDR module be represented by a set $\emptyset_{LDR} = [\emptyset_{HAM}, \emptyset_{DMRM}]$. The LDR module is trained using batch manner where the final loss is derived by combining the loss of HAM and DMRM for a given b^{th} batch of sample K, which can be represented as,

$$[Loss_{LDR}]^b = Loss_{HAM} + Loss_{DMRM}$$
$$= \frac{1}{K}\sum_{i=1}^{K} Loss_i(hm_i, p_{hm_i}) + \frac{1}{K}\sum_{i=1}^{K} Loss_i(p_i, gt_i) \quad (3.14)$$

The problem can be treated as an optimization problem where the total loss, $Loss_{Final}$ is minimized which can be represented as follows,

$$\underset{\emptyset_{LDR}}{argmin} \; [Loss_{LDR}]^b \tag{3.15}$$

For optimizing the Equation 3.15, the Adam optimiser [170] is used. The Algorithm 3.2 describes step by step procedures adopted to optimize the Equation 3.15.

Algorithm-3.2 Training and Optimizing the LDR Module

Input: Volume of frames, $V = \{v_1, v_2, \ldots\ldots, v_t, \ldots\ldots v_N\}$. The N represents total number of sequences in a video dataset.

Ground-Truth Crowd Head Maps: The set $HM = \{hm_1, hm_2, \ldots\ldots, hm_N\}$ represents ground truth crowd head maps for N frames.
Ground-Truth Crowd Density Maps: The set $GT = \{gt_1, gt_2, \ldots\ldots, gt_N\}$ represents ground truth crowd density maps for N frame sequences.
Parameters: $learning\ rate\ (\eta)$, $Learnable\ parameters\ (\emptyset_{LDR})$, $momentum$.
Initialisation: $Max\_Iter = 2500$, b$atch_Size, kernel$ regularize value of $L_2 - norm = 0.01$ $counter = 1, \eta = 0.001$, and the patience parameter of early stopping is set to 30.
Output: Optimize the LDR Module
While *early_stopping* or $counter = Max\_Iter$ is fulfilled, do
  For each batch *b = 1 to* $\left\lceil \frac{N}{Batch_Size} \right\rceil$do
    For each sample i in batch bt do
      1. Obtain the multiscale feature F_i.
      2. Find the loss of HAM i.e., $[Loss_{HAM}]^i$ and also loss of DMRM i.e., $[Loss_{DMRM}]^i$ using Equation 3.12 and 3.13 respectively.
    end for
    3. Obtain the mean of cumulative of loss for the given batch b using Equations 3.15.
    4. Find the gradients of the loss $[Loss_{LDR}]^b$ using [169].
    5. Obtain Cumulative History of gradients, and optimize Equation 3.15 by updating
    LDR's learnable parameters $\emptyset_{LDR}$, using Adam [170] optimizer.
  End For
  6. $counter += 1$
End While

3.3.1.3 Working of the Weakly-Supervised GCCR module.

The GCCR module contains an MLP module designed to map the global density map attribute to ground-truth global crowd count. By adopting the process of Savner *et al.* [45], the main objective behind GCCR is to consider the global density property and adopt a weakly-supervision methodology to regression to crowd count instead of density

maps and improve the performance. The global density property is obtained by summing all elements of the crowd density map. Let the global density property for all the N samples be represented by a set $GD = \{gd_1, gd_2, \ldots, gd_N\}$. For i^{th} sample, the global density property can be obtained by using the following equation.

$$gd_i = \sum_{j=1}^{Size} p_{i_j} \tag{3.16}$$

Now, the gd_i is inputted to the MLP. The MLP contains two hidden dense connection layers having 256 and 512 neurons. The second hidden layer is densely connected with the output layer with a single neuron. The output layer predicts global crowd counts. The ground-truth global crowd count has to be generated, which is obtained from ground-truth density maps. Let the set $GC = \{gc_1, gc_2, \ldots, gc_N\}$ represent the set of ground truth global crowd counts. For i^{th} sample, the ground truth global crowd count gc_i can be obtained using Equation 3.17.

$$gc_i = \sum_{j=1}^{Size} gt_{i_j} \tag{3.17}$$

Let the predicted global crowd count be represented by using a set $PGC = \{pgc_1, pgc_2, \ldots, pgc_N\}$. Let θ_{GCCR} represents all the trainable parameters of the GCCR module. This paper utilizes the squared error to capture the loss between pgc_i and gc_i for i^{th} sample, which can be represented as in Equation 3.18.

$$[Loss_{GCCR}]^i = Loss_{GCCR_i}(pgc_i, gc_i) = (p_i - gt_i)^2 \tag{3.18}$$

The optimization is done using batch manner. For a given b^{th} batch of sample K, the loss is represented as in Equation 3.19.

$$[Loss_{GCCR}]^b = \frac{1}{K}\sum_{i=1}^{K}(p_i - gt_i)^2 \tag{3.19}$$

Now, the loss can be minimized by solving the following optimization problem using the Adam optimizer [170],

$$\underset{\theta_{GCCR}}{argmin}\ [Loss_{GCCR}]^b \tag{3.20}$$

The Algorithm-3.3 describes step by step procedures adopted to optimize the Equation 3.20.

Algorithm-3.3 Training and Optimizing the GCCR Module

Input: Global Density Property, $GD = \{gd_1, gd_2, \ldots., gd_N\}$. The N represents total number of sequences in a video dataset.

Ground-Truth Global Crowd Count: The set $GC = \{gc_1, gc_2, \ldots., gc_N\}$ represents ground truth global crowd counts for N frames.
Parameters: $learning\ rate\ (\eta)$, $Learnable\ parameters\ (\theta_{GCCR})$, $momentum$.
Initialisation: $Max\_Iter = 2500$, $Batch\_Size, kernel$ regularize value of $L_2 - norm = 0.01$ $counter = 1, , \eta = 0.001$, and the patience parameter of early stopping is set to 30.
Output: Optimize the GCCR Module
While *early_stopping* or $counter = Max\_Iter$ is fulfilled, do
  For each batch b = *1 to* $\left\lceil \frac{N}{Batch\_Size} \right\rceil$ do
    For each sample i in batch bt do
      1. Feed forward the features to the output layer.
      2. Obtain the output and find the loss of GCCR i.e., $[Loss_{GCCR}]^i$ using Equation 3.18.
    end for
    3. Obtain the mean of cumulative of loss for the given batch b i.e., $[Loss_{GCCR}]^b$ using Equation 3.19.
    4. Find the gradients of the loss $[Loss_{GCCR}]^b$ using [169].
    5. Obtain Cumulative History of gradients, and optimize Equation 3.20 by updating GCCR's learnable parameters θ_{GCCR}, using Adam [170] optimizer.
  End For
  6. $counter += 1$
End While

For optimizing the two-deep models, i.e., LDR and GCCR, we have used backpropagation [169] with Adam optimizer [170]. The learning rate η is set to 0.001. We have used L_2 kernel regularization and set the value of the regularization parameter to 0.01.

3.3.2 Experimental Setup

The two deep models have been coded from scratch in python using Keras libraries and TensorFlow. Several computing nodes of the Param-Shivay supercomputer

are used to execute the code. The models are implemented separately in a cascading fashion. The models are trained by setting the total number of epochs to 1000. The batch size for the Mall [57], Venice [4], and UCSD [59] is set to 64, 8, and 64, respectively. The regularization is of utmost importance to avoid overfitting or underfitting. In the proposed model, the L_2 regularization is used to regularize all the convolution layers' kernel weights and early stopping to halt the training of the proposed models. The regularization parameter value is set to 0.01 and the patience parameter of early stopping to 24. The learning rate η, momentum of batch normalization, regularized parameter of L_2, decay rates for first (β_1) and second moment (β_2) of Adam optimiser[170] are initialized to 0.001, 0.95, 0.01, 0.9, and 0.999, respectively. The maximum iteration was to 2000.

3.3.3 Results Analysis and Discussion

3.3.3.1 The Venice Dataset

The LDR module is implemented separately with different sizes of the input set. We have limited the experiments with frames' volume (FV) of sizes 3, 4, 5, and 6. The comparison of results with six state-of-the-art approaches are illustrated in Table 3.7. The values mentioned in bold letters in Table 3.7 are the best in the table. On the Venice dataset, the LDR module with $FV = 3$, $FV = 4$, $FV = 5$ and $FV = 6$ achieve <MAE and RMSE> of <31.27, 47.63>, <30.52, 37.77>, <28.29, 39.50> and <24.58, 34.76> respectively. The LDR module with $FV = 6$ performs better than other sizes of the volume of frames. The LDR ($FV = 6$) + GCCR module achieves MAE and RMSE of 18.59 and 27.52 for global crowd counting. The single-image crowd counting models like MCNN[27], Switch-CNN[85], CSR-Net[94] and ECAN[4] achieves <MAE, RMSE> of <145.4, 147.3>, <52.8, 59.5>, <35.8, 50.00> and <20.5, 29.9> respectively. The ECAN[4] performs better as compared with other single image crowd counting models.

There are only two video-based crowd counting models implemented on the Venice dataset: AMS-CNN[33] and TMCMS-ST Atrous-Net[34]. These two models achieve <MAE and RMSE> of <23.64, 28.75> and <49.17, 58.22>. However, the proposed video-based crowd counting model performs better than the AMS-CNN and ECAN[4]. Hence, we can conclude that the proposed model effectively handles the challenges and results in achieving better performance as far as Venice is concerned. Figure 3.12 and Figure 3.13 show plotting of predicted versus ground-truth crowd count on the Venice dataset[4] by the LDR and LDR+GCCR modules, respectively.

Table 3.7: Comparisons of Results of several approaches on Venice

Approaches/Models	**MAE**	**RMSE**
Switch-CNN[85]	52.80	59.50
MCNN[27]	145.4	147.3
CSR-Net[94]	35.8	50.00
ECAN[4]	20.50	29.90
AMS-CNN[33]	23.64	28.75
TMCMS-ST Atrous-Net[34]	49.17	58.22
LDR ($FV = 3$)	31.27	47.63
LDR ($FV = 4$)	30.52	37.77
LDR ($FV = 5$)	28.29	39.50
LDR ($FV = 6$)	24.58	34.76
LDR ($FV = 6$) + GCCR	**18.59**	**27.52**

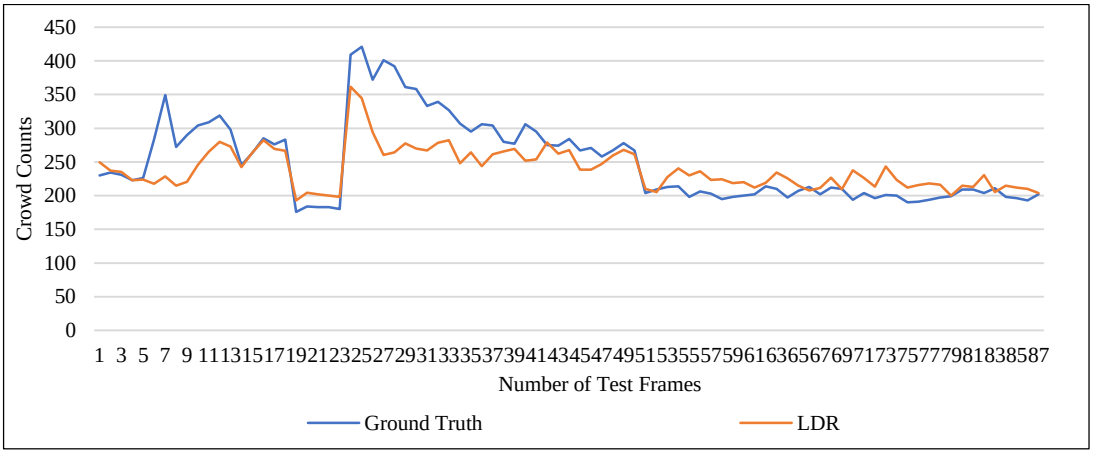

Figure 3.12: Predicted versus Ground-truth Crowd Counts of LDR-Module on the Venice Dataset [4]

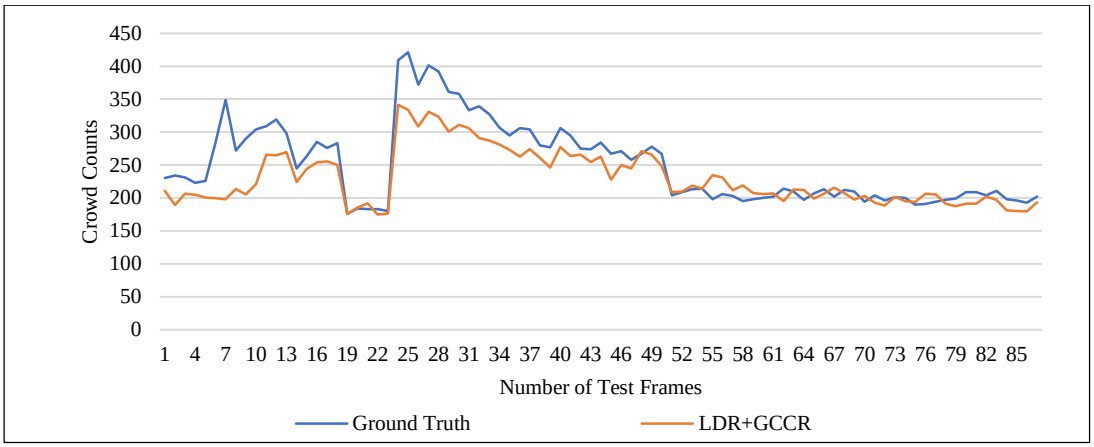

Figure 3.13: Predicted versus Ground-truth Crowd Counts of LDR+GCR-Module on the Venice Dataset [4]

3.3.3.2 The Mall Dataset

The results obtained on the Mall dataset [57] are compared with 16 state-of-the-art approaches. The comparison of results is mentioned in Table 3.8, where the values in bold letters are highest in the table. The proposed LDR module with $FV = 3$, $FV = 4$, $FV = 5$ and $FV = 6$ achieve <MAE and RMSE> of <3.12, 4.02>, <3.02, 3.74>, <2.52, 3.21> and <2.40, 3.02> respectively. Among different sizes of FV, the LDR module performs better with $FV = 6$. So, we have used the output of LDR with $FV = 6$ as input to the GCCR module. The GCCR module (i.e., LDR ($FV = 6$)+GCCR) achieves MAE and RMSE of 2.34 and 2.97, respectively. Figure 3.15 and Figure 3.16 show a line graph representing the predicted versus the ground-truth crowd count on the Mall dataset [57] by the proposed LDR ($FV = 6$) and LDR ($FV = 6$)+GCR modules, respectively.

The conventional models like Ridge regressor [57], kernel ridge regression (KRR) [58], gaussian process of regression (GPR) [59], and Count Forest [70] produce abysmal results as compared with the proposed LDR and LDR+GCCR module. Among the single image-based techniques like MCNN [27], CCNN [28], CNN-MRF [171], Faster R-CNN [172], DAL-SVR [173], and DIGCrowd [174], the DAL-SVR [173] performs better with MAE and RMSE of 2.4 and 9.57 respectively.

Table 3.8: Comparisons of Results of several approaches on the Mall dataset [57]

Approaches/Models	**MAE**	**RMSE**
Ridge Regression [57]	6.73	19.18
KRR [58]	6.61	18.85
GPR [59]	7.15	21.34
Count Forest [70]	5.75	10.88
CNN-MRF [171]	4.66	9.01
Faster R-CNN [172]	4.65	7.26
MCNN [27]	4.74	8.64
CCNN [28]	5.36	9.34
ConvLSTM-nt [12]	2.53	11.2
ConvLSTM [12]	2.24	8.5
Bidirectional ConvLSTM [12]	**2.10**	7.60
DAL-SVR [173]	2.40	9.57
DIGCrowd [174]	3.21	16.4
ST-CNN [32]	4.03	5.87
AMS-CNN [33]	2.47	3.08
DAL-SVR [173]	2.40	9.57
DIGCrowd [174]	3.21	16.4
TMCMS-ST Atrous-Net [34]	3.72	4.74
Proposed LDR ($FV = 3$)	3.12	4.02
Proposed LDR ($FV = 4$)	3.02	3.74
Proposed LDR ($FV = 5$)	2.52	3.21
Proposed LDR ($FV = 6$)	2.40	3.02
Proposed LDR ($FV = 6$)+GCCR	2.34	**2.97**

Among the video-based CCDE approaches such as AMS-CNN [33], TMCMS-ST Atrous-Net [34], ST-CNN [32], ConvLSTM [12], ConvLSTM-nt [12], Bidirectional ConvLSTM [12], R-DCNN [163], LDR, and LDR+GCCR, the Bidirectional-ConvLSTM [12] has the better MAE, i.e., 2.1 but the proposed LDR+GCCR approach achieves better RMSE of 2.97. The counting model should have low MAE, RMSE, and the difference should be minimal.

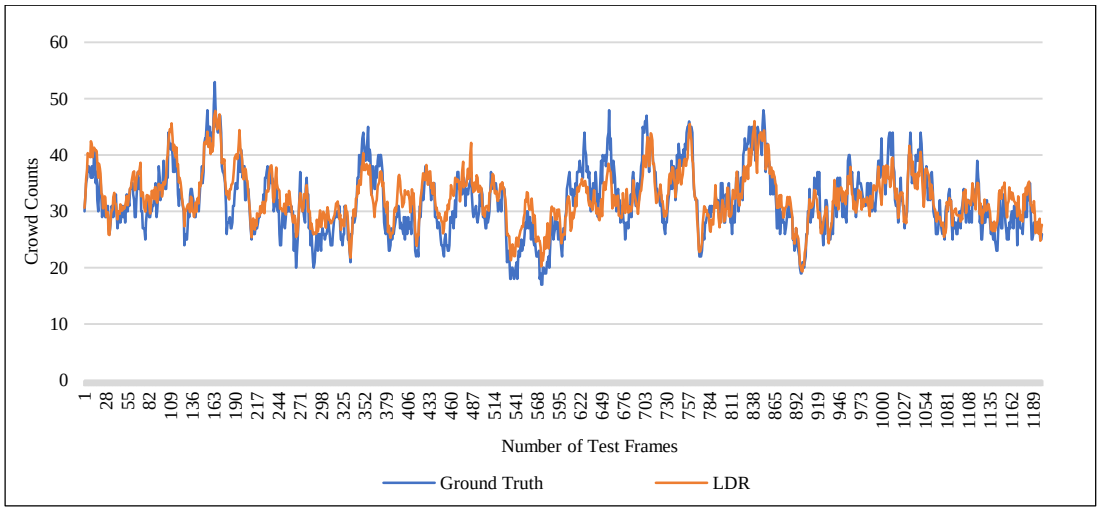

Figure 3.14: Predicted versus Ground-truth Crowd Counts of LDR-Module on the Mall Dataset [57]

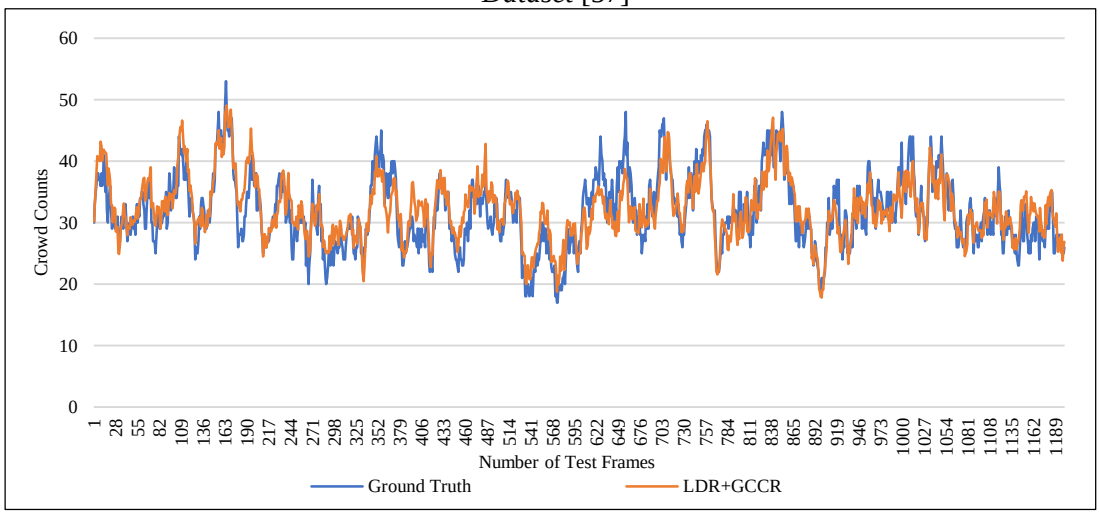

Figure 3.15: Predicted versus Ground-truth Crowd Counts of LDR+GCCR Module on the Mall Dataset [57]

Based on the above discussions, we can conclude that the proposed model provides better results than other listed video-based CCDE approaches by obtaining minimum RMSE and comparatively better MAE. So, the head attentive spatial-temporal features fusion model yields in comparatively better solution among all.

3.3.3.3 The UCSD Dataset

The comparison of results with other state-of-the-arts on the UCSD dataset [59] is illustrated in Table 3.9. The values mentioned in bold letters in Table 3.9 are the best in the table. The proposed LDR achieves <MAE, RMSE> of <2.85, 3.75>, <2.67, 3.73>,

<1.95, 2.37> and <1.47, 1.85> on $FV = 3, 4, 5\ and\ 6$ respectively. On the other hand, the LDR ($FV = 6$)+GCCR obtains MAE=1.45 and RMSE=1.84. We observe little improvement by LDR ($FV = 6$)+GCCR over the LDR module.

Table 3.9: Comparisons of Results of several approaches on UCSD

Model Name	**MAE**	**RMSE**
GPR [59]	2.24	7.97
Ridge regression [57]	2.25	7.82
CAR [60]	2.07	6.90
Density map + MESA [68]	1.70	NA
Count forest [70]	1.60	4.40
CCNN [28]	1.51	-
CrossCrowdNet [175]	1.60	3.31
Switch-CNN [85]	1.62	2.1
DAL-SVR [173]	1.29	2.10
ConvLSTM-nt [12]	1.73	3.52
ConvLSTM [12]	1.30	1.79
Bidirectional ConvLSTM [12]	**1.13**	**1.43**
AFP [177]	1.16	2.29
FCN-rLSTM [111]	1.54	3.02
MDMF-CC [90]	1.35	1.88
MS-GAN [176]	1.78	3.03
AMS-CNN [33]	1.46	1.82
Proposed LDR ($FV = 3$)	2.85	3.75
Proposed LDR ($FV = 4$)	2.67	3.73
Proposed LDR ($FV = 5$)	1.95	2.37
Proposed LDR ($FV = 6$)	1.47	1.85
Proposed LDR ($FV = 6$)+ GCCR	1.45	1.84

Among other techniques, the Bidirectional ConvLSTM [12] obtain MAE=1.13 and RMSE=1.43, which is the highest performance as far as Table 3.9 is concerned. The proposed model is placed in sixth and fourth positions in Table 3.9 in terms of MAE and RMSE. Nevertheless, the performance difference between the proposed model and other techniques [12], [90], [173], [177] is less. Figure 3.16 and Figure 3.17 show plotting of predicted versus ground-truth crowd count on the UCSD dataset by the LDR and LDR+GCCR modules, respectively.

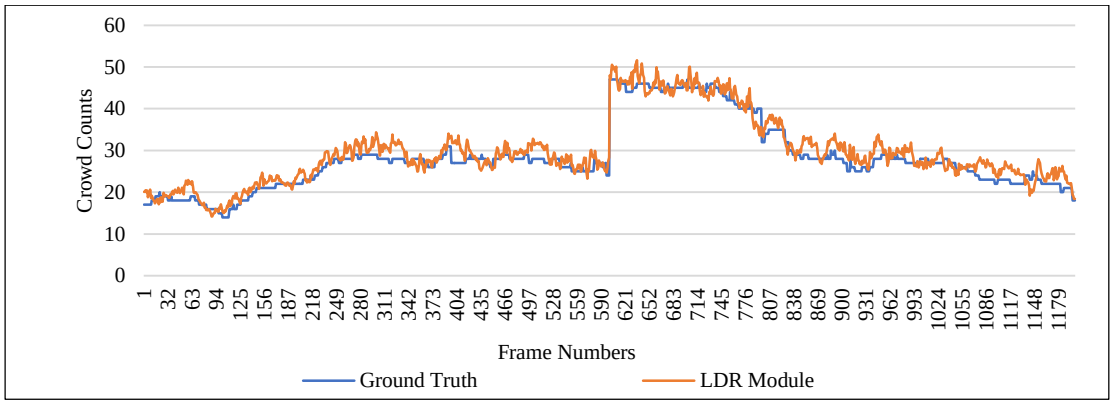

Figure 3.16: Predicted versus Ground-truth Crowd Counts of LDR-Module on the UCSD Dataset [59]

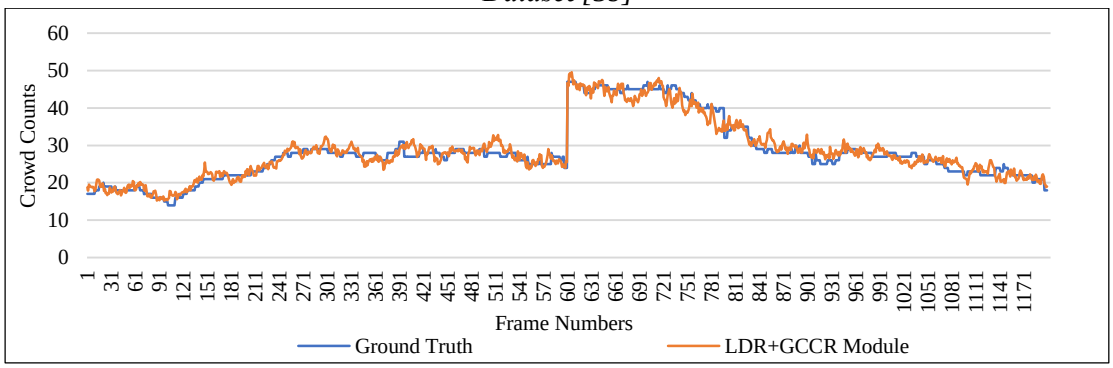

Figure 3.17: Predicted versus Ground-truth Crowd Counts of LDR+GCCR Module on the UCSD Dataset [59]

3.3.3.4 Ablation study

An ablation study on the proposed cascaded model is conducted to study the impact of each module of the cascaded deep model on video-based crowd counting. The proposed is split into the following independent modules based on the combination of different each stream of the LDR module with the GCR module,

- LDR (FV=6) (Column1): It contains the Column1 of LDR followed by HAM and DMRM.
- LDR (FV=6) (Column2): It contains the Column2 of LDR followed by HAM and DMRM.
- LDR (FV=6) (Column3): It contains the Column3 of LDR followed by HAM and DMRM.

- LDR (FV=6) (WO-HA): It contains the MF-MSFFM followed by DMRM. Here, the main objective is to observe the model's response without head attention (WO-HA).
- LDR (FV=6) (WO-MLFI): It contains all the modules of LDR but without multi-layer feature inclusion (WO-MLFI)
- LDR (FV=6) (WO-MLFI -HA): It contains all the modules of LDR but without multi-layer feature inclusion and head attention (WO-MLFI-HA)
- LDR (FV=6): The exact LDR module
- LDR (FV=6) + GCCR: The proposed cascaded deep model.

The training procedures are the same as described in the first paragraph of Section 5. The following Table 3.10 shows the comparison of results of different modules of the proposed model.

Table 3.10: Comparisons of results for ablation study on the different datasets

Several Module Name	**The Mall Dataset**		**The Venice Dataset**		**The UCSD Dataset [59]**	
	MAE	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
LDR (FV=6) (Column1)	2.62	3.36	31.81	46.57	3.49	4.64
LDR (FV=6) (Column2)	2.80	3.50	35.41	50.75	3.40	4.91
LDR (FV=6) (Column3)	3.20	3.97	28.03	47.11	4.73	7.01
LDR (FV=6) (WO-HA)	2.91	3.58	38.16	55.22	3.77	5.13
LDR (FV=6) (WO-MLFI)	2.68	3.39	36.55	53.69	3.48	4.71
LDR (FV=6) (WO-MLFI -HA)	2.97	3.76	40.34	59.21	2.83	3.45
LDR (FV=6)	2.40	3.02	24.58	34.76	1.47	1.85
LDR (FV=6) + GCCR	**2.34**	**2.97**	**18.59**	**27.52**	**1.45**	**1.84**

On the Venice dataset [4], the Column1, the Column2, and the Column3 obtains <MAE, RMSE> of <31.81, 46.57>, <35.41, 50.75> and <28.03, 47.11> respectively. The Column3 of the proposed model performs better other two columns. The without head attention modules like LDR (FV=6) (WO-HA) achieve MAE=38.16 and RMSE=55.22, which performs less compared to LDR (FV=6) and LDR (FV=6) + GCCR. The LDR (FV=6) (WO-MLFI) and LDR (FV=6) (WO-MLFI -HA) obtain <MAE, RMSE> of <36.55, 53.69> and <40.34, 59.21> respectively. Now, by comparing these modules'

performances with the proposed module, it can be pointed out that individual modules could not achieve the performance we get by combining them.

The Column1, Column2 and Column3 of the proposed model achieve <MAE, RMSE> of <2.62, 3.36>, <2.80, 3.50> and <3.20, 3.97> respectively on the Mall dataset [57]. Among the three columns, Column1 performs better. The without head maps model, i.e., LDR (FV=6) (WO-HA), gets MAE and RMSE of 2.91 and 3.58 and performs comparatively low concerning LDR and LDR + GCCR. Hence, the head attention module becomes important to minimize the background influence and improve performance. The three-column architecture without multi-layer feature inclusion, i.e., LDR (FV=6) (WO-MLFI), achieves MAE and RMSE of 2.68 and 3.39, respectively. On the other hand, the LDR (FV=6) (WO-MLFI -HA) module achieves MAE=2.97 and RMSE=3.76. Whereas the proposed LDR (FV=6) and LDR (FV=6) + GCCR achieves <MAE, RMSE> of <2.40, 3.02> and <2.34, 2.97> respectively. So, it can be summarized that the manifold multiscale fusion, head attention modules, and the GCCR modules are requirements for better performance.

In Table 3.10, the three columns individually achieve <MAEs, RMSEs> of <3.49, 4.64>, <3.40, 4.91> and <4.73, 7.01>. Similar kinds of performances have been achieved by the modules LDR (FV=6) (WO-HA) and LDR (FV=6) (WO-MLFI). However, the LDR (FV=6) (WO-MLFI -HA) module obtains MAE=2.83 and RMSE=3.45. Unlike Mall [57] and Venice [4], the individual modules' performance is quite less than the proposed models' performance on the UCSD dataset [59]. On the other hand, the proposed model achieves MAE=1.45 and RMSE=1.84, which are near similar values as compared with other state-of-the-art approaches.

3.4 Conclusion

This chapter proposed two video-based CCDE approaches using deep learning techniques. The first model, AMS-CNN, enhanced feature quality and minimized the effect of cluttered background by infusing deep spatial foreground features. At the same time, the second model overcomes the issues of varying crowd shape and background details in crowd videos. Also, the second model used a weak supervision learning mechanism to minimize the error caused due to point-level annotations. Both the models were evaluated on three publicly available datasets: the Mall [57], the Venice [4], and the UCSD [59], and the performance metrics used were MAE and RMSE. The first model achieved <MAE, RMSE> of <2.47, 3.08>, <23.64, 28.75> and <1.46, 1.82> on the Mall [57], the Venice [4], and the UCSD [59] respectively while the second deep model achieved <MAE, RMSE> of <2.34,2.97>, <18.59, 27.52> and <1.45, 1.84> on the Mall [57], the Venice [4], and the UCSD [59] respectively outperforming the state-of-the-art methods. The second model outperforms the first model by extracting multiscale spatial-temporal features, considering both local-global crowd properties, and adopting weak supervision to overcome point-level annotation errors, which are not addressed in the first model. An extensive ablation study of both models was performed to show the efficacy of different modules.

This chapter discussed the work done on the task of CCDE for analyzing crowd videos using deep learning techniques. The next chapter will discuss the proposed work for the CCA.