An Efficient Low-Light Bi-Directional Traffic Flow Estimation based on Instance Segmentation using Mask R-CNN and a Counting GRU

Dale Dantis*, Dhanush D Shekar* and Dinesh Naik*
*National Institute of Technology, Karnataka, Surathkal

***Abstract*—This paper aims to implement an accurate approach for vehicle counting in videos using Mask R-CNN and an autonomous counting methodology using a GRU.Current implementations of Line of Interest counting are fairly weak in dense traffic scenarios. Hence, to alleviate the need of a tracking algorithm, we use a time-series model. Instead of using a multi-target tracking procedure to track and count each vehicle, a counting network termed the counting Gated Recurrent Unit Memory (cGRU) network is developed to analyze bi-directional CF and count vehicles in successive video frames. Before vehicle detection and feature extraction, each input frame is preprocessed using the process of Histogram Equalization to improve its low-light nighttime performance. Based on the cGRU Output, the traffic flow parameters are found using the Accumulator. The model is trained and tested on the UA-DETRAC Dataset.**

***Index Terms*—Histogram Weighing Module, Histogram Equalization, Mask R-CNN, Instance Segmentation, LSTM, GRU, Vehicle Counting, Bi-directional Traffic Flow, Vehicle Detection**

I. INTRODUCTION

In terms of road traffic, India is one of the busiest countries on earth. In 2017, the automotive sector in the south Asian country surpassed that of the United States to become the world's fourth-largest.However, with this boom in the number of automobiles, there is a proportional growth in the number of traffic problems.

Nowadays, one of the most popular research topics is traffic control. The importance of traffic control may be seen in the information provided regarding road closures, accidents, and other types of road disturbances,which helps the appropriate authorities to respond accordingly.

Estimation of traffic flow parameters, such as volume, density, and speed, in real-time is a crucial requirement for some traffic management and guidance systems. In dense traffic scenarios, estimating these values usually demands vehicle counting, which is still a challenge.

Vehicle detection efficiency is regarded as the most important factor in vehicle counting and traffic monitoring in general. Over the years many researchers have proposed various vehicle detection methods as traffic monitoring has grown popular. Several detection methods are based on the distinction between the moving foreground and the static background scene; these methods are known as motion-based methods. The difference between consecutive frames can be used to detect the vehicle objects, as discussed in [1].

There exist two mainstream methods of Vehicle Detection using Computer Vision,namely, LOI or Line of Interest and ROI or Region of Interest.Estimation of traffic volume is more accurately determined using an LOI method, while ROI methods usually perform better at estimating traffic density.

A novel LOI counting framework for dense traffic scenarios is proposed based on this hypothesis, which includes a salient vehicle detector using MASK R-CNN, RSWHE based histogram equalization for image quality enhancement, a bi-directional traffic flow feature (Count features), a counting network(cGRU), and a traffic flow parameter accumulator.

To help the model operate more accurately in a wider variety of scenarios, we employ an advanced version of Histogram Equalization called the Recursively Separated and Weighted Histogram Equalization for Brightness Preservation and Contrast Enhancement(*RSWHE*). Histogram Equalization aims to enhance the contrast of the image. This particular process proves helpful in dark environments since it helps to enhance the brighter parts of the image while simultaneously darkening the darker parts of the image. Many of the current Histogram Equalization methods face a particular issue with the mean brightness being shifted. Our proposed model implements a Histogram Equalization sub-model that enhances the contrast while almost completely alleviating the mean-brightness shift.

The following are the study's major procedures and contributions:

1) For vehicle detection, an instance segmentation is performed using Mask R-CNN [2] which generates a pixel-wise mask for the detected objects along with bounding box predictions. This is more accurate than the bounding boxes detection as used as no overlapping of the bounding box occurs [3].
2) To enhance the contrast of the image, Histogram Equalization, or more specifically Recursively Separated and Weighted Histogram Equalization(RSWHE-M). This manages to enhance the infrequent intensity levels

of the while normalizing the frequent intensity levels using a normalized power law function.

3) For busy traffic conditions, the use of a count feature(CF) is presented to treat the vehicle flow, moving in both directions as a whole unit. The CF feature is essential in the operation and extraction of bi-directional dense traffic flow features, allowing the traffic flow to be described as a whole rather than as individual vehicles.
4) A Counting Gated Recurrent Unit,that is a time-series RNN,is modeled as a series classification framework, using the extracted CF feature as inputs to the model. The proposed cGRU is the key to avoiding the use of multi-target tracking in this method. A similar approach is proposed in [4] where a cLSTM model, which is computationally more expensive, is made use of for the counting model. Also a YOLOv3 framework is used for vehicle detection which is a bounding-box based object detection model, which has a bounding-box overlapping problem in dense traffic scenarios.
5) A parameter estimation model is created based on the LOI counting findings for estimating the traffic flow characteristics of volume, density, and speed. Figure 1 gives a summarized flow of the whole approach.

II. Related Work

Fundamentally, the approaches used for vehicle counting methods are broadly classified into sensor-based and vision-based approaches. The sensor-based approaches basically make use of real time data obtained from hardware components that need to be physically installed. The monitoring devices used are inductive lops, magnetic sensors and pneumatic tubes which are less accurate and sometimes might disregard some slow moving vehicles during vehicle counting [5] - [6]. However on the contrary, the coming of age computer vision methods are more robust and versatile while still facing some problems in complex traffic scenarios like dense traffic, occluded vehicles, unstable camera etc.

There are two forms of Vehicle Counting and Traffic Flow estimation, namely Sensor-based Counting or Vison-based Counting. Sensor-based Counting, as the name suggests, makes use of different kinds of sensors such as magnetic sensors to track Vehicle Count. Although, this would be more accurate, it would be extremely time-consuming to set up, and an even bigger investment, with the requirement of regular maintenance Vision-based Counting ,on the other hand requires no such sensors and is relatively very simple to set, with very low maintenance requirement/. The downfall of the latter vision method is that it is just not as accurate as its counterpart, with its performance worsening considerably when the number of vehicles to count increases.

Another method described in [7], builds a model for static background using several samples of the background scene. This model is used to remove the backdrop and allow for the detection of foreground objects. The authors of [8] propose an adaptive bounding box algorithm for vehicle detection, where the background model is created using Gaussian Mixture Modeling (GMM), and then the vehicle objects are detected in the foreground

All of the previous methods produce good results, but there are still some issues that affect detection accuracy. The dynamic background environment, for example, has a significant impact on motion-based detection approaches, as the concept of these methods is focused mostly on vehicle motion. The prediction accuracy may be harmed by motion caused by rain, snow, or even changes in lighting. For appearance-based techniques, a thorough understanding of the object's properties is required. In order to acquire decent detection results, a lot of work is required to get accurate and reliable feature extraction [9]. Deep learning approaches address all of these challenges and are now regarded the cutting-edge technology for computer vision applications.

In general all the LOI counting approaches implemented in the past don't really focus on dense traffic scenarios. As seen in [10], [11] and [12], where vehicle tracking algorithms are used to perform vehicle counting using LOI, non-dense traffic conditions are preferred for the best performance. Similarly, in LOI methods where background subtraction procedures are used, as observed in [13], [14], the models implemented are essentially insignigicant when a dense traffic scenario is considered while the performance for a small group of vehicles is optimal. There are two major causes for this. For instance, earlier LOI counting systems typically required detecting and tracking each car in high traffic circumstances, which is a time-consuming and complicated task. Secondly, when the foreground targets are close together, the background extraction process in TSI becomes difficult. Due to these challenges, only a few approaches concentrate on the LOI counting problem in congested traffic settings. To begin with, tracking mechanisms track each and every vehicle on the road to keep track of their positions to estimate traffic count. As is obvious, the performance reduces as the traffic increases, all while the computational resource requirments increase. Secondly, in the case of dense traffic scenarios, the foreground targets are near each other, throwing the ability of the model to extract the background off.

In the case of Histogram Equalization, many forms of methods were considered to circumvent the mean-brightness shift error. [15] aims to retain the mean brightness while maximising entropy while also targeting consumer electronics. [16] talks about RSWHE, which has been currently implemented in our model, while [17] and [18] evaluate and compare the different methods of Histogram Equalization.

III. Methodology

A. Histogram Equalization

Various methods have been used to implement the concept of Histogram Equalization while also circumventing its disadvantages. Global Histogram Equalization(GHE) uses the Cumulative Distribution Function(CDF) to modify the Probability Distribution Function(PDF) such that it approaches a uniform distribution. This method seemed to distort the overall image's brightness. To avoid this, various methods

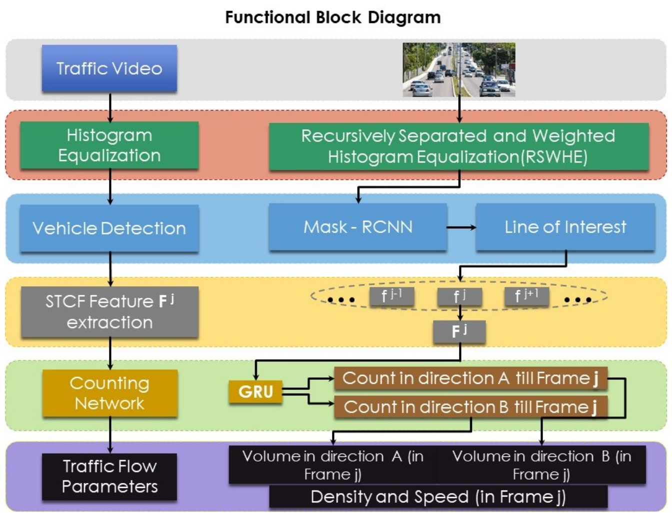
Functional Block Diagram
Traffic Video
Histogram Equalization
Recursively Separated and Weighted Histogram Equalization(RSWHE)
Vehicle Detection
Mask - RCNN
Line of Interest
STCF Feature F j extraction
f j-1
f j
f j+1
F j
Counting Network
GRU
Count in direction A till Frame j
Count in direction B till Frame j
Traffic Flow Parameters
Volume in direction A (in Frame j)
Volume in direction B (in Frame j)
Density and Speed (in Frame j)

Fig. 1. Block-Diagram Flow of the Complete Model

were used to recursively divide the image to obtain a more granular control over the brighter and darker parts, namely RMSHE(Recursive Mean Separate Histogram Equalization), RSIHE(Recursive Sub-Image Histogram Equalization), each dividing the histograms of the images recursively into sub-histograms based on the Mean and the Median of the image respectively and perform GHE but over the sub-images instead of the complete image. These methods perform better than the original method but are relatively poor at limiting noise and maintaining the integrity of the original image. Another method called MMBEBHE(Minimum Mean Brightness Error Bi-Histogram Equalization) was proposed, which divided the image at each intensity value for every iteration. The Average Mean Brightness Error(AMBE) is calculated for each iteration till a Global minimum is found. This method, quite obviously, proves to be very computationally taxing.

The above models perform better than GHE when reducing the mean-brightness shift between the Input and the Output image, but they never modify the image data before performing Histogram Equalization. Our current model implements a modified version of RMSHE: RSWHE or Recursively Separated and Weighted Histogram Equalization for Brightness Preservation and Contrast Enhancement. In this method, the input image is weighted using a Normalized Power-Law function. As Figures 5, 6, 7 mention, RSWHE performs the best in almost all cases. The visual effect of RSWHE vs the original and the MATLAB in-built $HistEq$ with their respective PDFs function can be seen in Figure 4.

There are 3 modules that makeup RSWHE:-

- **Histogram segmentation module:-** Recursively divides the image histogram into sub-histograms based on the mean intensity of the Image.
  For every recursion ,assuming P_m is the mean value which divides input image into two parts P_L and P_U then P_L and P_U is be defined in 1-

$$P = P_L \cup P_U \tag{1}$$

where

$$P_L = \{p(i,j)|\, p(i,j) \ \leq \ P_D \,\forall\, p(i,j) \ \in P\}$$

and

$$P_U = \{p(i,j)|\, p(i,j) \ > \ P_D \,\forall\, p(i,j) \ \in P\}$$

and

$$P_m = \frac{(\sum_{i=0}^{M-1} i.pdf(P_i))}{(\sum_{i=0}^{M-1} pdf(P_i))} \tag{2}$$

Where sub-image P_L is composed by the and sub-image P_U is composed by the $\{p_e, P_{e+1}, ...M-1\}$. Now let's probability density function for sub-images $P_L and P_U$ respectively are 3 and 4-

$$pdf_L(P_k) = n_k/N_L, where\ k = 0,1,2....D. \tag{3}$$

and

$$pdf_U(P_k) = n_k/N_U, where\ k = D+1, D+2...M-1. \tag{4}$$

- **Histogram weighing module:-**
  After performing r recursions, we know that 2^r sub-histograms $(H_i^r(P) - \{1 \ \leq \ i \ \leq \ 2^r\})$ are formed. Let the original PDF and weighted PDF for each sub-histogram $(H_i^r(P))$ be $p(P_k)$ and $p_w(P_k)$ respectively.

$$p_w(P_k) = p_{max}\left(\frac{p(P_k) - p_{min}}{p_{max} - p_{min}}\right)^{\alpha_i} + \beta \tag{5}$$

Where, $k : (L_i \leq k \leq U_i)$
p_{min} and p_{max} are the original histogram's minimum and maximum probability value respectively,
α_i- accumulative probabilty value.

$$\alpha_i = \sum_{k=L_i}^{U_i} p(P_k) \tag{6}$$

β is used to control the mean brightness and contrast enhancement of the output image in 7.

$$\beta = p_{max}.|P_M - P_G|/(P_{max} - P_{min}) \tag{7}$$

P_{max} and P_{min} are the highest and least intensity levels of the input image P.

- **Histogram equalization Module:-**
  The resultant modified and normalized PDF is separately equalized for all 2^r sub-histograms by using GHE using the modified sub-histograms as Inputs.

ALGORITHM:

1) Find the median/mean intensity of the input image using equation 2.
2) Divide the images based on the median/mean intensity using equation 1.
3) Repeat 1 and 2 for the number of recursions but now use the sub-image as the I/P image.
4) Weigh the Original pdf using the Normalized Power Law Function using equation 5.
5) Normalize the weighed pdf.
6) Use this pdf as the I/P to the pdfs of the sub-images (scaling them to appropriate values).
7) Apply GHE on these sub-images.
8) Combine the changed sub-images back together.

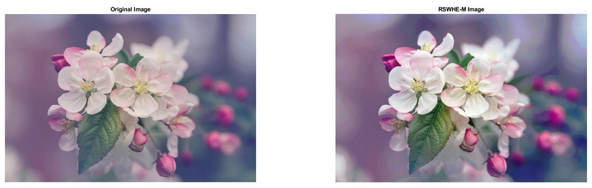

Fig. 2. Original vs RSWHE-M

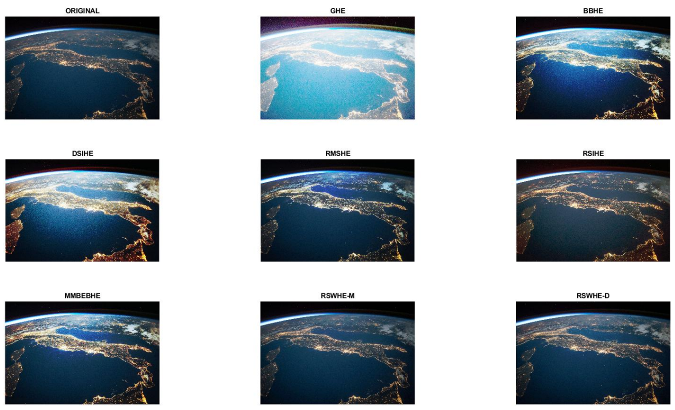

Fig. 3. Earth with different Histogram Equalizations

B. Mask R-CNN

Mask R-CNN is an advanced implementation of the Faster R-CNN model, which has the added benefit of instance segmentation, which is particularly helpful in dense traffic scenarios. Mask R-CNN is a segmentation framework for object instances that was originally released in 2017 [4]. It was created by adding a parallel channel to the Faster R-CNN in order to provide a segmentation mask for each identified object in addition to the bounding box and the class label as seen in

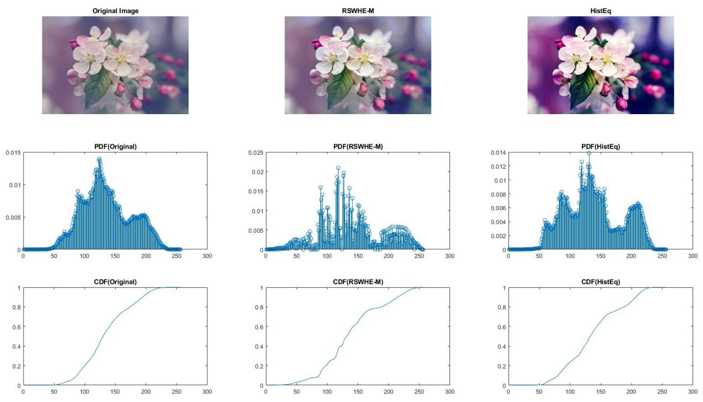

Fig. 4. Original vs MATLAB HistEq vs RSWHE-M

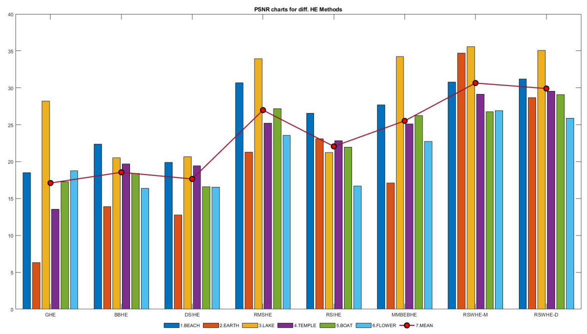

Fig. 5. PSNR for each Histogram Equalization for each Image

Figure 8. Mask R-CNN has two constituent stages, the first step is to use the Region Proposal Network (RPN) to forecast which regions may contain objects. The RoIAlign layer, which distinguishes Mask R-CNN from its predecessors, is used to align pixels to pixels. The object bounding box, class label, and object mask are then generated in parallel using two extended R-CNN heads. A mask encodes the spatial layout of an input object. Convolutions' pixel-to-pixel correspondence can readily handle retrieving the spatial structure of masks, unlike

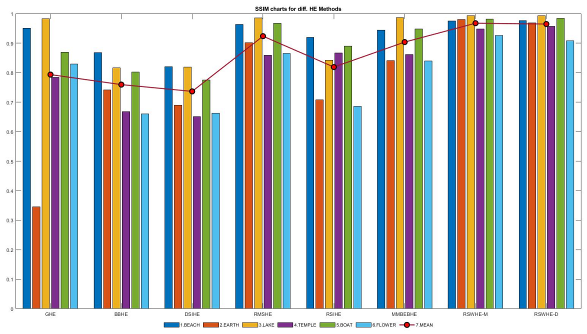

Fig. 6. SSIM for each Histogram Equalization FOR each Image

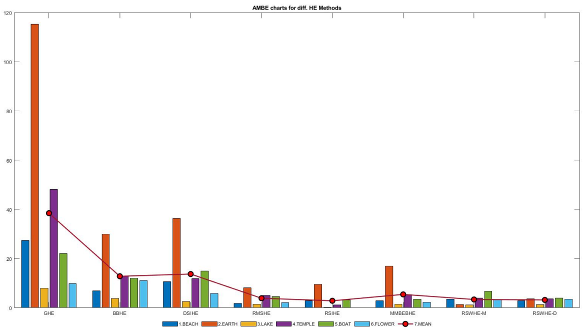

Fig. 7. AMBE for each Histogram Equalization FOR each Image

class labels or box offsets, which are generally condensed into short output vectors by fully connected (FC) layers as see in Figure 9.

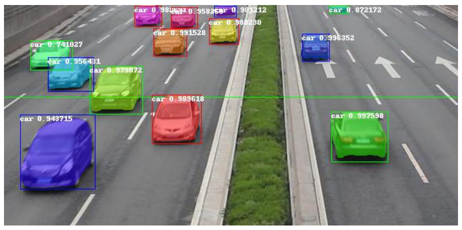

Fig. 8. Instance segmentation by Mask R-CNN on MVI_40191

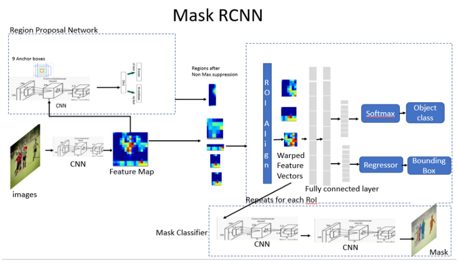

Fig. 9. Architecture of the Mask RCNN

C. Spatio-Temporal Count Features

The bounding boxes obtained from the Mask R-CNN is then manipulated to obtain the Count Features that represent the vehicles that have a high probability of crossing the Counting Line at any given frame. Hence Count Features depend on space and time.

$$bb_i = (y_{1i}, x_{1i}, y_{2i}, x_{2i}), where\ i = 1, 2, 3...N_d, \tag{8}$$

Where,
y_1= the top y coordinate,
y_2= the bottom y coordinate,
x_1 = the left x coordinate,
x_2= the right x coordinate,
N_d= the number of detected vehicles in frame.

Let

$$D^j = bb_1^j, bb_1^j, bb_1^j, ...bb_{Nd}^j$$

$$bb_i^j = (y_{1i}^j, y_{1i}^j, y_{1i}^j, y_{1i}^j), i = 1, 2, ...N_d$$

The Spatial attribute of the CF is then obtained using equation 9:-

$$\delta_i^j = \frac{(y_i^j + h_i^j)}{2} - c_y, (i = 1, 2, ...N_d) \tag{9}$$

where c_y = Counting Line

Out of N_d vehicles detected in each frame j, the top N_h vehicles of the highest probabilities of crossing the Counting Line are selected. This is done by obtaining the absolute values of the Spatial attribute and arranging them in ascending order. The first N_h elements create the N_h dimensional Spatial attribute vector as shown in equation 10.

$$f^j = [\delta_1^j, \delta_2^j, ..., \delta_{N_h}^j] \tag{10}$$

An appropriate value of N_h must be obtained so as to achieve good accuracy. Increasing N_h beyond a reasonable value brings in unwanted features and hence can negatively affect the accuracy of the model. On the other hand, decreasing N_h beyond a reasonable value might bring too little feature and hence tend to lose track of counting cars. The Counting Feature is then obtained using equation 11:-

$$F^j = [f^{j-n}, ..., f^{j-1}, f^j, f^{j+1}, ..., f^{j+m}] \tag{11}$$

where F_j = Counting Feature for the frame j.

The settings of n and m must ensure that the frames between $(j-n)$ and $(j+m)$ can cover each vehicle's whole crossing operation for all frames within a specified period. Too many frames will draw unnecessary data few will not gather the required data for optimal counting performance, similar to the balance required to identify N_h.

D. LOI Counting Framework

The LOI architecture consists of three modules:- The Counting Features, the cGRU network, and Accumulator To avoid the use of complex tracking algorithms, a time-series Recurrent Neural Network is used as shown in Figure 10. Shuang et al. [4] used a counting LSTM to track the traffic flow as a whole. Our proposed model makes use of a counting GRU. The GRU is a simpler version of the LSTM and can hence operate relatively more quickly and efficiently. The CF is provided as an Input to the cGRU, which in turn classifies it into three classes, each representing a state of traffic flow. To train cGRU, the extracted CF is used to derive a variable called class using equation 12:

$$class_k^j = sgn(\delta_k^j) - sgn(\delta_k^{j-1}), \tag{12}$$

where, s_k^j is the Spatial Attribute for the k^{th} vehicle and the j^{th} frame. The sgn function is 13:

$$sgn(x) = \begin{cases} 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \\ -1, & \text{if } x < 0 \end{cases} \tag{13}$$

The $class_k^j$ is then used to find the Training Label using equation 13.

$$O^j = \begin{cases} 1, & \text{if } \sum_k class_k^j = 0 \\ 2, & \text{if } \sum_k class_k^j > 0 \\ 3, & \text{if } \sum_k class_k^j < 0 \end{cases} \tag{14}$$

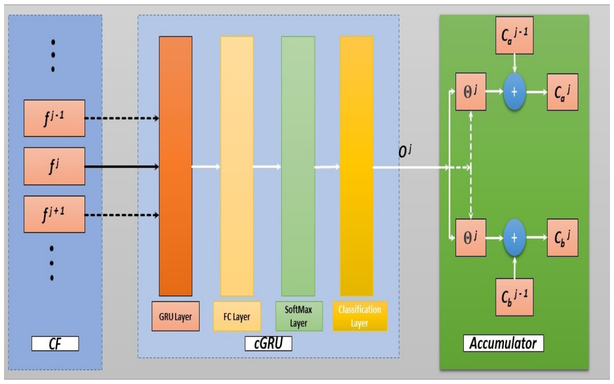

Fig. 10. Complete LOI Counting Flow

E. cGRU Architecture

The cGRU module consists of 5 layers(in order):-

- A Sequence Layer that takes the Count Features as input.
- A GRU layer with 500 hidden units processes the Count Feature and computes the
- 3 Fully Connected Layers processes the GRU Output
- A SoftMax Layer
- A Classification Layer with 3 Output classes
- The model layers and their configurations are given in Table I and Figure 11.

TABLE I
MODEL ARCHITECTURE

	Name	Type	Activations	Learnables
1	Sequence Input(Sequence input with 3 dimensions)	Sequence Input	3	-
2	gru (GRU with 500 hidden units)	GRU	500	InputWeights (1500x3) RecurrentWeights (1500x500)
3	Fully Connected Layer (3 fully connected layers)	Fully connected	3	Weights (3x500) Bias (3x1)
4	softmax	Softmax	3	-
5	Classoutput (crossentropyex loss)	Classification Output	3	-

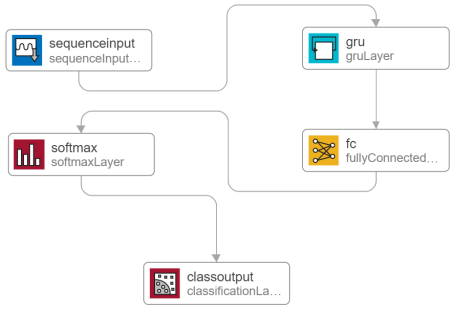

Fig. 11. cGRU Architecture Model

F. Accumulator

This module is used to compute the traffic flow parameters based on the cGRU Output. To compensate for the relatively large amount of data generated in class - 1 a dynamic threshold analysis method is employed.

The Spatial Attributes of the N_d detected vehicles are classified into two sets: A and B, each representing a direction. Hence:

$$A^j = b_1^j, b_2^j, ..., b_\mu^j \tag{15}$$

$$B^j = b_1^j, b_2^j, ..., b_\nu^j \tag{16}$$

Where μ and ν are the number of vehicles in direction A and B in the j_{th} frame, respectively.

$$dp_j = min(\eta_i^j), (i = 1, 2, ..., I), \forall \quad \eta_i^j \geq 0, \tag{17}$$

$$dn_j = max(\eta_i^j), (i = 1, 2, ..., I), \forall \quad \eta_i^j < 0, \tag{18}$$

where n_i^j is the spatial attribute per frame of the CF. I can be considered as μ if O^j = 2 or ν if O^j = 3. These values can be used to obtain the vector in equation 19:

$$d_j = [dp_j, dn_j], \tag{19}$$

This vector represents the closest vehicles to the counting line. One vehicle has crossed the counting line, while the other one has not. The Euclidean distance E_j is the difference between d_j and d_{j-1}.

$$\theta_j = \alpha.ave(E_{j-w}), (w = 1, 2, ..., n), \forall \quad O_{j-w} = 1, \tag{20}$$

α is considered to be 2. N is the number of useful frames, while E_{j-w} represents the difference between d_{j-w} and d_{j-w-1} for O_{j-w} = 1. And hence, the traffic count in each direction per frame is given using equation 21:-

$$\begin{cases} C_A^j = C_A^{j-1} + 1, & \text{if } O^j = 2 \text{ } and \text{ } E_j > \theta_j, \\ C_B^j = C_B^{j-1} + 1, & \text{if } O^j = 3 \text{ } and \text{ } E_j > \theta_j, \end{cases} \tag{21}$$

The Volume of the Traffic per frame in both Direction A and B is given using equation 22 and 23:-

$$V_A^j = C_A^j \times \frac{3600.R}{j}, \tag{22}$$

$$V_B^j = C_B^j \times \frac{3600.R}{j}, \tag{23}$$

Furthermore, the density, equivalent to the number of vehicles per kilometer, can be calculated using equations 24 and 25:

$$Des_A^j = \frac{D_A^j.L_p}{l.L_a}.1000, \tag{24}$$

$$Des_B^j = \frac{D_B^j.L_p}{l.L_a}.1000, \tag{25}$$

where D_a^j and D_b^j are the number of vehicles in each direction per frame, In the close-up region, l is the pixel length of the road segment, while the L_p and L_a are the pixel length and corresponding actual length, respectively.

The traffic speed can be calculated using equations 26 and 27:

$$S_A^j = \frac{V_A^j}{Des_A^j}, \tag{26}$$

$$S_B^j = \frac{V_B^j}{Des_B^j}, \tag{27}$$

IV. Discussion and Results

A. Dataset

The proposed counting model is evaluated on the UA-DETRAC dataset, which consists of 10 hours of footage shot using a Canon EOS 550D camera in 24 different sites across Beijing and Tianjin, China. These films were shot at a frame rate of 25 frames per second (fps) and at a 960 x 540 pixels resolution. The dataset has bidirectional traffic videos covering scenarios like head-light glare, rainy roads, nighttime traffic, misaligned camera, dense traffic, etc. Training on this dataset ensures that our proposed counting methodology is trained and tested under various circumstances, making it robust in the process. From the dataset, four movies were chosen for our experiments, which are given in Table II.

B. Histogram Equalization

The process of dividing the images into sub-images using its Histogram is done 8 times using the mean intensity of the image/sub-image. There are 256 sub-images over which the Histogram Equalization process is run.

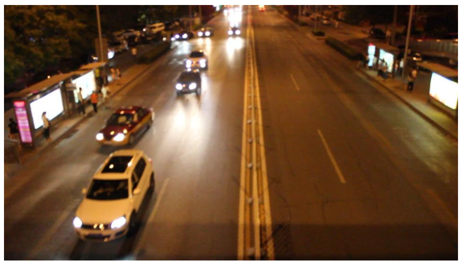

Fig. 12. Original Image (Low Light)

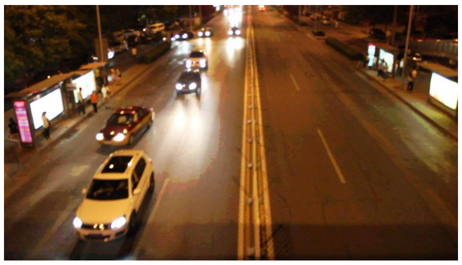

Fig. 13. RSWHE-M Image (Low Light)

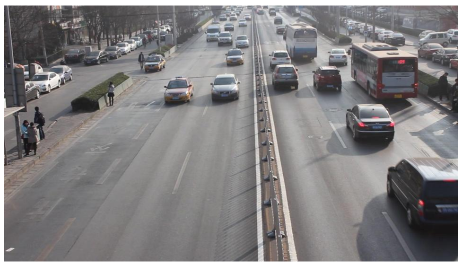

Fig. 14. Original Image (Day Light)

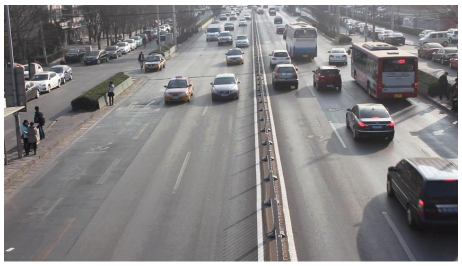

Fig. 15. RSWHE-M Image (Day Light)

TABLE II
PERFORMANCE WITH AND WITHOUT RSWHE-M

Video	GT of Vehicle Count	Vehicle Count without RSWHE	Vehicle Count with RSWHE	Scenarios
MVI_39851	36	18	38	Night, Occlusion, Unstable Camera
MVI_40191	286	300	300	Cloudy, Unstable Camera
MVI_20034	34	216	36	Sunny, Unstable Camera

As seen in the given Table II, the presence of RSWHE has a noticeable impact on the accuracy of the model, especially at nighttime. Though, in some cases, the output remains unchanged. Figures 12 and 13 show the frames before and after application of RSWHE in night time while Figures 14 and 15 show that for daylight traffic images.

C. Input Image vs. Output Image Errors

The Equations for the plots from Figure 5 - Figure 7 are given as follows:

- *Absolute Mean Brightness Error (AMBE):* It's a term used to describe how much brightness is preserved in a processed image. It can be defined using equation 28:

$$AMBE(P,Q) = |P_M - Q_M| \tag{28}$$

Where, P_M represents the mean of the input image $P = \{p(i,j)\}$ and Q_M represents mean of the output image $Q = \{q(i,j)\}$.

- *Structured Similarity Index (SSIM):* It is used to measure the similarity between two images.

$$SSIM(p,q) = \frac{(2\mu_p\mu_q + C_1)(2\sigma_{pq} + C_2)}{(\mu_p^2 + \mu_q^2 + C_1)(\sigma_p^2 + \sigma_q^2 + C_2)} \tag{29}$$

Here,
μ_p, μ_q−Mean of Image P and Q respectively.
σ_p, σ_q−Standard deviation of image P and Q respectively.
C_1, C_2 are the contents.

- *Peak signal to noise ratio (PSNR):* PSNR is calculated using using equation 31:
Consider an image with $M \times N$. Mean Squared Error given using equation 30,

$$MSE = \frac{\sum_{i=1} M \sum_{j=1} N |P(i,j) - Q(i,j)|^2}{M \times N} \tag{30}$$

$$PSNR = 10log_{10}\frac{(L-1)^2}{MSE} \tag{31}$$

D. Vehicle Detection

As mentioned earlier, the open-source Mask R-CNN model was used to perform vehicle detection due to its ability to perform instance segmentation by pixel-pixel masking. The model was loaded with the pre-trained weights it was trained for on the COCO dataset. The model was further customized to our needs by detecting and segmenting only the objects belonging to the classes: cars, trucks, buses, bicycles, and motorcycles. Further, the model was configured to run on GPU with a TensorFlow backend and Keras API on Python 3. The batch size of the model was set to 1 image per GPU. The model had a total of 64,158,584 parameters with 64,047,096 Trainable parameters and 111,488 Non-trainable parameters. The whole implementation was carried out on Google Collab using a Tesla-K80 GPU and an Intel(R) Xeon(R) CPU @ 2.30 GHz with 32 GB of RAM.

For vehicle detection, the input frames were further cropped to zoom in on the image in order to eliminate the unwanted detections of vehicles by Mask R-CNN, which were farther from the desired line of interest[LOI] and hence significantly redundant. The movies used and their close-up regions considered are listed in Table III. An instance of the close-up operation being performed is showcased in Figures 16 - 17 and 18 - 19 for MVI_20034 and MVI_39851 respectively. It roughly took around one second for all the vehicle detections at every frame. Once all the detections were made, the Mask R-CNN outputs obtained included the bounding box coordinates used to calculate the spatial attributes and further the Count Features as given in equations a and b, respectively.

TABLE III
DETAILS OF THE DATASET

Video	Original Resolution	Resized Resolution	Close up Region
MVI_20034	960 x 540	350 x 700	[100:450 , 200:900]
MVI_39851	960 x 540	375 x 620	[25:400 , 180:800]
MVI_40191	960 x 540	300 x 620	[100:400 , 180:800]
MVI_20061	960 x 540	200 x 540	[50:250 , 360:900]

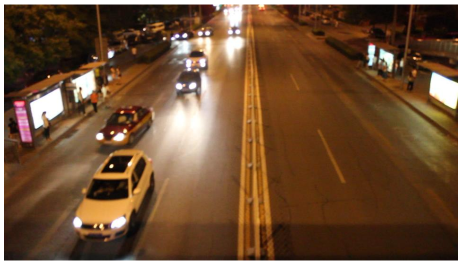
Fig. 16. Original Image (Low Light)

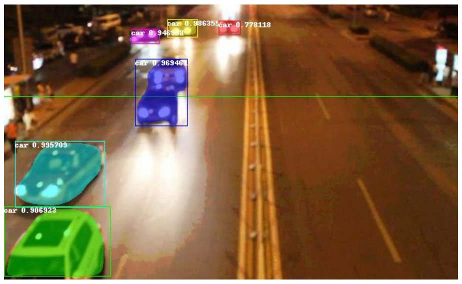

Fig. 17. Zoomed and Detected Image (Low Light)

E. Vehicle Detection Performance

The performance of vehicle detection by Mask R-CNN on the close-up area of the traffic images is mentioned in Table IV. The Mask R-CNN seems to perform better on the stable camera video capture of MVI_40191, while it seems to get slightly affected for occluded and unstable video capture of MVI_39851. The XML annotation files provided with the

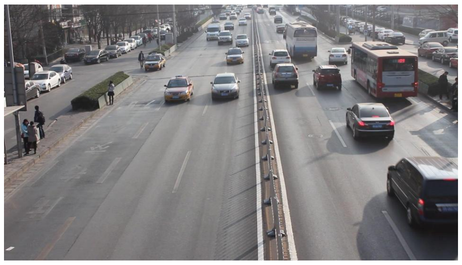

Fig. 18. Original Image (Day Light)

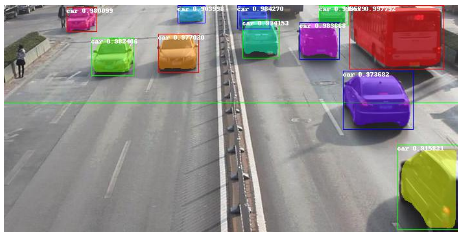

Fig. 19. Zoomed and Detected Image (Day Light)

dataset were used to compare the detected vehicles with the ground truth to calculate detection accuracy at each frame.

TABLE IV
MASK R-CNN DETECTION ACCURACY

Video	Number of frames	Accuracy	Scenarios
MVI_38951	1415	90%	Night, Occlusion, Unstable Camera
MVI_40191	2496	96%	Cloudy, Unstable Camera
MVI_20034	801	97.5%	Sunny, Unstable Camera

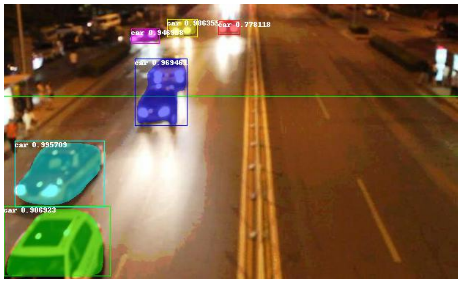

Fig. 20. Image with all Detections

F. The Count Feature extraction and cGRU model training

The cGRU network was trained using the Adam optimize with a learning rate of 0.001. To prevent gradients fron exploding, the gradient threshold was set to one. The model was trained for 25 epochs, and there were 101 iterations every epoch. Sequence length was set to six as a total of six frames were used in every Count Feature. The number of inputs was equal to $N_h(3)$, and the classifier output layer had three classes

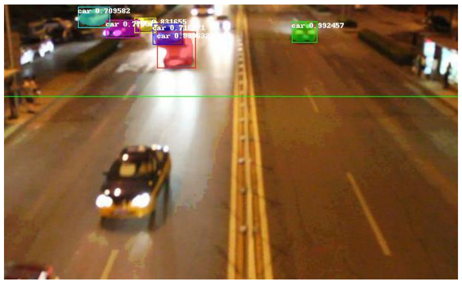

Fig. 21. Image with one Detection missing

to predict.Figure 22 and 23 shows the training progress plots of the cGRU for 2525 Iterations.

The training data had a total of 2155 sequences, out of which 1995 frames belonged to class1 while only a mere 60 sequences belonged to class 2 and class 3 as a whole, with class 2 having 24 samples and class 3 having 36 samples. To overcome this data imbalance, a series of data augmentation was done exclusively on class2 and class 3 samples, including scaling and inversion of the existing samples. Finally, the dataset was amplified to contain 15,000 samples, with 4032 samples belonging to classes 2 and 3. The Count Feature is made up of Spatial attributes with n = 2 and m = 3. Hence the the total dimension of the Count Feature per frame is (2 + 3 + 1) = 6. The N_h was considered to be as 3. These values provided the most optimal data for the cGRU to train on since the model's accuracy was very low with different permutations of n,m, and N_h.

The training and testing are done in MATLAB, with an Intel Core i7-7700HQ and an NVIDIA GeForce GTX 1080 Max-Q. The accuracy of the cGRU model without the data-augmented input was around 65%. Only by augmenting the data did it create a better model.

The Accuracy is calculated using the below formula:-

Where the Difference = Ground Truth(GT) - Estimated Count(EC)

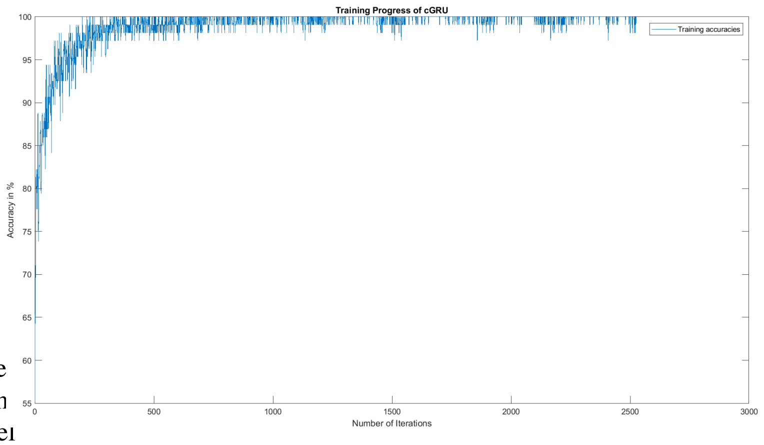

Fig. 22. (GRU Training Accuracy)

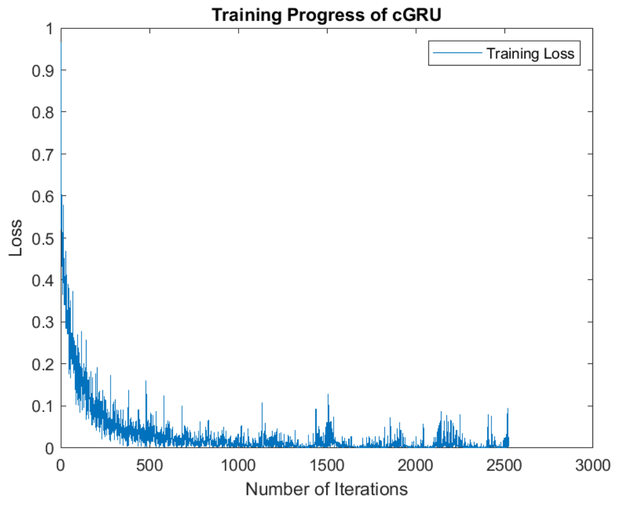

Fig. 23. (GRU Training Loss)

G. cGRU vs. cLSTM

As explained by the results shown in Table V, the cGRU is considerably faster and more efficient when handling small packets of data. This is because cGRU is a simplistic form of cLSTM and hence has trouble handling more significant amounts of data.

TABLE V
GRU Performance vs LSTM Performance

Time-Series Model	Training Time(s)	Testing Time (MVI_40191) (s)	Training Accuracy (%)	Testing Accuracy (MVI_20034) (%)
cGRU	125.988	70.601	99.7347	94.117
LSTM	7181.203	70.738	99.2481	69.857

H. Traffic Flow Parameters

Estimation of traffic flow parameters involved the use of equations 22 - 27 for calculating Volume,Density and the average Speed of vehicles in both Direction A and Direction B.

Figure 24 shows the count of vehicles in both the directions over time in the video(MVI_40191). Figure 25 shows the estimated density of vehicles per kilometer in both the directions, while Figures 26 and 27 show the estimation plots of Volume of vehicles per hour and average Speed of vehicles respectively in both the directions.

Table VI consists the compiled results of traffic flow parameters estimated for different videos of the UA-DETRAC dataset.

I. Comparison with Other Methods

Table VII tabulates the average performance of various State of the Art methods in Vehicle Counting. As depicted, the proposed model performs less accurately and slower compared to the LSTM-based LOI Counting Model. This can

TABLE VI
Traffic Flow Evaluation

Data	Directions	Count	Volume	Speed	Density
MVI_39851	A	11	700	28.0057	25
	B	27	1718	52.7016	33
	Both	38	2418	80.7072	58
MVI_40191	A	232	8389	69.34	13
	B	68	2459	144.25	18
	Both	300	10848	106.795	33
MVI_20034	A	10	1134	53.4365	22
	B	26	2948	126.3044	24
	Both	36	4081	89.87	46

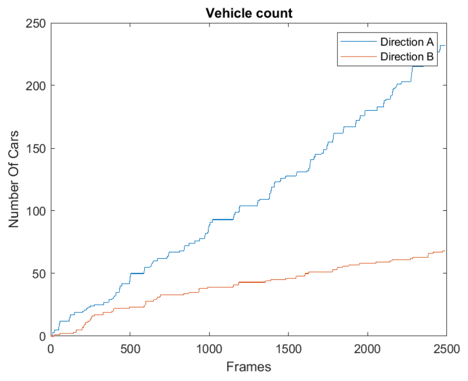

Fig. 24. Count of the vehicles in both Directions(MVI_40191)

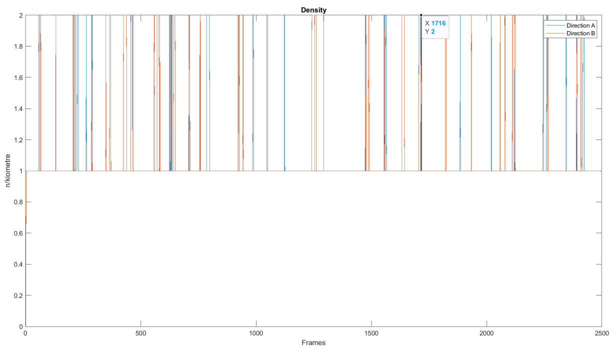

Fig. 25. Density of the vehicles in both Directions(MVI_40191)

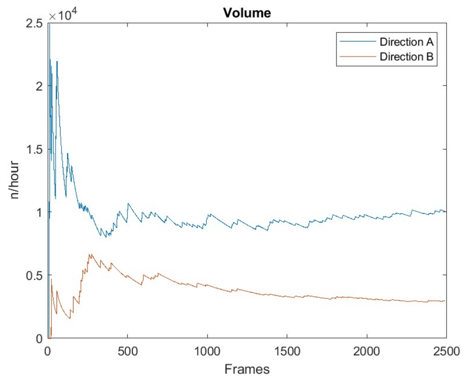

Fig. 26. Volume of the vehicles in both Directions(MVI_40191)

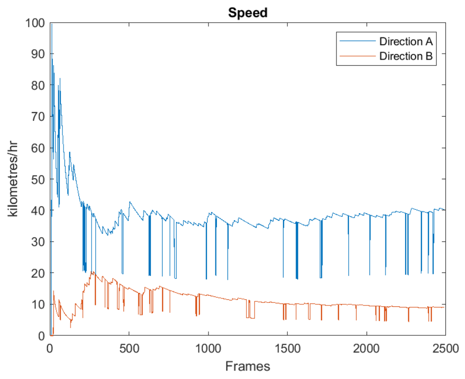

Fig. 27. Speed of the vehicles in both Directions(MVI_40191)

TABLE VII
PERFORMANCE AGAINST OTHER COUNTING MODELS

Model	**MVI_20034**		**MVI_40191**	
	Accuracy	Speed (fps)	Accuracy	Speed (fps)
FasterRCNN+CMOT	83.64%	9.21	84.40%	9.70
FasterRCNN+H^2T	87.27%	8.76	88.07%	9.05
compACT+CMOT	89.09%	3.79	90.36%	3.62
compACT+H^2T	90.01%	3.02	91.28%	3.11
YOLOv3+CMOT	94.55%	11.96	93.12%	11.50
YOLOv3+H^2T	92.73%	13.05	94.04%	12.77
ESCNN+KLT	92.73%	26.33	92.66%	25.53
FastBS	90.91%	99.23	91.74%	98.87
YOLOv3+cLSTM	98.18%	49.23	99.08%	48.81
Proposed Model	**94.117%**	**3.318**	**95.104%**	**3.327**

be attributed to the fact that the the current implementation of the model is not robust, despite performing well on multiple datasets. The Speed of the model in its current iteration is also of concern here. While the GRU outperforms the LSTM by a considerable margin(as seen in Table V), the preprocessing and detection network ends up being the bottleneck. The Mask R-CNN takes 0.2s to detect per frame, while the RSWHE takes 0.1s to process per frame. The main scope in further research would be optimization schemes to enhance the performance of the model.

TABLE VIII
PERFORMANCE OF THE COMPLETE MODEL

Videos	GT (Vehicle Count)	EC (Vehicle Count)	Accuracy (%)	Speed(fps)
MVI_39851	36	38	94.444	3.324
MVI_40191	286	300	95.104	3.327
MVI_20034	34	36	94.117	3.318
MVI_20061	31	159	0	3.318

V. CONCLUSION

In this paper an effective approach for vehicle detection and counting using a LOI based approach is presented, which involves the use of instance segmentation for vehicle detection, RSWHE for low-lit images,and a counting based GRU model. The pixel-pixel masking of Mask R-CNN mitigates the issues of bounding-box overlapping problems faced in the previously proposed methods for vehicle detection in dense scenarios. Furthermore to make this process more robust the input video frames are processed with RSWHE before using Mask-RCNN for better low light performance which is exemplified by an increase in vehicle counting accuracy of 50% to 94.44% before and after the application of RSWHE for MVI_39851. Also a GRU based counting model is implemented, which takes in the Count features as an input and feeds the classified output to an accumulator which keeps count of the vehicles. Since there is no tracking algorithm, the only computationally intensive task is the Training of the cGRU. Ideally, this alleviates the need for a high-end Compute Unit(GPU/TPU) to work in real time.

The proposed model seems to show good accuracies when it comes to certain Datasets. It also performs worse in some other scenarios. By far two of the biggest flaws in the current implementation of the model, is its consistency and speed, both of which can be enhanced with further progress.

REFERENCES

[1] H. Jia-feng, "Vehicles detection based on three-frame-difference method and cross-entropy threshold method," *Computer Engineering*, 2011.

[2] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," 2018.

[3] Z. Al-Ariny, M. A. Abdelwahab, M. Fakhry, and E.-S. Hasaneen, "An efficient vehicle counting method using mask r-cnn," pp. 232–237, 2020.

[4] S. Li, F. Chang, and C. Liu, "Bi-directional dense traffic counting based on spatio-temporal counting feature and counting-lstm network," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, 2020.

[5] B. Liu, Q. Li, D. Chen, and H. Sun, "Pattern recognition of vehicle types and reliability analysis of pneumatic tube test data under mixed traffic condition," vol. 3, pp. 44–47, 2010.

[6] E. Sifuentes, O. Casas, and R. Pallas-Areny, "Wireless magnetic sensor node for vehicle detection with optical wake-up," *IEEE Sensors Journal*, vol. 11, no. 8, pp. 1669–1676, 2011.

[7] S. Gupte, O. Masoud, R. Martin, and N. Papanikolopoulos, "Detection and classification of vehicles," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 3, pp. 37 – 47, 04 2002.

[8] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1773–1795, 2013.

[9] S. Kamkar and R. Safabakhsh, "Vehicle detection, counting and classification in various conditions," *IET Intelligent Transport Systems*, vol. 10, no. 6, pp. 406–413, 2016.

[10] R. Ke, Z. Li, J. Tang, Z. Pan, and Y. Wang, "Real-time traffic flow parameter estimation from uav video based on ensemble classifier and optical flow," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 54–64, 2019.

[11] R. Ke, Z. Li, S. Kim, J. Ash, Z. Cui, and Y. Wang, "Real-time bidirectional traffic flow parameter estimation from aerial videos," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 4, pp. 890–901, 2017.

[12] Z. Dai, H. Song, X. Wang, Y. Fang, X. Yun, Z. Zhang, and H. Li, "Video-based vehicle counting framework," *IEEE Access*, vol. 7, pp. 64460–64470, 2019.

[13] Z. Gao, R. Zhai, P. Wang, X. Yan, H. Qin, Y. Tang, and B. Ramesh, "Synergizing appearance and motion with low rank representation for vehicle counting and traffic flow analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2675–2685, 2018.

[14] Y. Zhang, Z. Chihang, and Q. Zhang, "Counting vehicles in urban traffic scenes using foreground time-spatial images," *IET Image Processing*, vol. 11, pp. 61–67, 03 2017.

[15] C. Wang and Z. Ye, "Brightness preserving histogram equalization with maximum entropy: a variational perspective," *IEEE Transactions on Consumer Electronics*, vol. 51, no. 4, pp. 1326–1334, 2005.

[16] M. Kim and M. G. Chung, “Recursively separated and weighted histogram equalization for brightness preservation and contrast enhancement,” *IEEE Transactions on Consumer Electronics*, vol. 54, no. 3, pp. 1389–1397, 2008.

[17] D. U. Singh, J. Kumre, and R. Singh, “Histogram based image enhancement techniques: A survey,” *INTERNATIONAL JOURNAL OF COMPUTER SCIENCES AND ENGINEERING*, vol. 5, 06 2017.

[18] O. Patel, Y. P. S. Maravi, and S. Sharma, “A comparative study of histogram equalization based image enhancement techniques for brightness preservation and contrast enhancement,” *CoRR*, vol. abs/1311.4033, 2013.