Low Light Video Enhancement

A Project Report

submitted by

GOKUL MOHANRAJ

in partial fulfilment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

&

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

JUNE 2022

THESIS CERTIFICATE

This is to certify that the thesis titled **Low Light Video and Image Enhancement**, submitted by **Gokul Mohanraj**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology and Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. A N Rajagopalan
Project Guide
Dept. of Electrical Engineering
IIT Madras, 600 036

Place: Chennai

Date: 20 June, 2022

ACKNOWLEDGEMENTS

I would like to express my gratitude towards my project guide, Prof. A. N. Rajagopalan for giving me the opportunity to work on this project. His extensive knowledge and experience in this field has been a great source of inspiration. I would also like to acknowledge Aakanksha, my PhD mentor and guide of this project and I am grateful for her support.

I would like to take this opportunity to specifically thank my faculty advisor, Dr. Anjan Chakraborty, who helped me quite a lot.

I would like to thank my parents for their encouragement and unwavering support throughout my life. Finally, I would also like to thank my friends for extending their help and support at all times.

ABSTRACT

KEYWORDS: Low-light, Convolutional Recurrent Neural Networks, Gamma correction, U-Net, audio signal processing

Images and videos make up a significant part of the digital world. Illumination is a key factor in ensuring quality specifically for videos. Although significant research exists for image enhancement, the quality suffers greatly when it comes to videos due to lack of temporal consistency. This task hasn't received much attention as collecting pair-wise low and normal light videos is almost an impossible task.

In this project, we have looked at low-light image conversion algorithms and extending it to videos while ensuring temporal consistency using the idea of parameter sharing. This was used to create a dataset of 143 paired low-light and normal light videos along with their corresponding audio features as a wav file. Additionally, a RGB-to-RGB low to normal light video conversion algorithm was implemented using existing 3D U-Net architectures. The model took low-light videos as input and produced the corresponding normal light videos as output while maintaining temporal consistency.

The last step of this research was to introduce another parallel branch to also allow for audio signal processing. This was implemented using Convolutional Recurrent Neural Networks (CRNNs) in tandem with the existing U-Net architecture. The key idea is to explore if audio signals add valuable information to the low-light video enhancement task. Comparable results were produced both models warranting for further research and studies into the topic.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

LLIE	Low-light Image Enhancement
LLVE	Low-Light Video Enhancement
CMOS	Complementary Metal-oxide Semiconductor
CLAHE	Contrast Limited Adaptive Histogram Equalization
DRV	Dark RAW Video
AVE	Audio Visual Event
SGD	Stochastic Gradient Descent
PSNR	Peak Signal-to-Noise Ratio
SSIM	Structural Similarity
MABD	Mean Absolute Brightness Differences
MSE	Mean Square Error

NOTATION

α, β	linear transformation parameters
γ	gamma correction parameter
w_0, w_1, w_2	weights assigned to parameters
μ_x	mean of Image signal x
σ_x^2	variance of image signal x
σ_{xy}^2	covariance of image signals x and y
br^t	brightness of image frame t

CHAPTER 1

INTRODUCTION

This chapter is an introduction about images and videos, specifically the low-light videos and the current models that go into conversion from one to the other. The specific differences between low-light images and videos and ones in normal setting are discussed along with challenges in converting one from the other. A quick review of the previous works and an introduction to the idea proposed is also mentioned.

1.1 Introduction to Images and Videos

Images and videos are an integral part of the world we live in today. Most camera systems in the world are looking to develop state of the art technologies in hardware and software for camera and imaging models. However, there are many challenges that go into creating a model that can produce clear and sharp visuals. The modern digital camera uses an aperture that opens on the click of a button allowing light to pour into the lens. Electronic equipments capture the incoming light rays and convert it into electrical signals using either a charged coupled device (CCD) or a CMOS image sensor. There are numerous challenges that come with this system like noise and blur, resolution, etc. Specifically, here we are dealing with environments that have low-lights. In such settings, image acquisition devices do not receive sufficient light sources resulting in outputs with low brightness and contrast of images. This poses a great challenge for other computer vision tasks that need to be performed. This has developed an entire new area of computer vision that deals with low-light enhancement algorithms. Low-light image enhancement (LLIE) and Low-light Video Enhancement (LLVE) deals with a set of algorithms that aim at improving the perception, interpret-ability and visual appeal of images and videos that were captured in settings where the illumination is significantly poor. The next section deals with the specific challenges that are present in images and videos that are shot in low-light settings.

1.2 Challenges with low-light settings

The main challenge with videos taken in extreme low-light conditions is the lack of visibility. Illumination between a low-light night and a sunny day can vary by more than 10 orders of magnitude. The obstacles are characterised by low dynamic range and high amounts of noise. This prevents computer vision algorithms from performing properly. In low-light settings, sensor noise cannot be ignored due to very poor SNR (signal-to-noise ratio). Noise in itself can come primarily from 2 sources, the recorded video and the measuring device. Enhancement algorithms attempt to model the signal and the noise and consequently enhance the image.

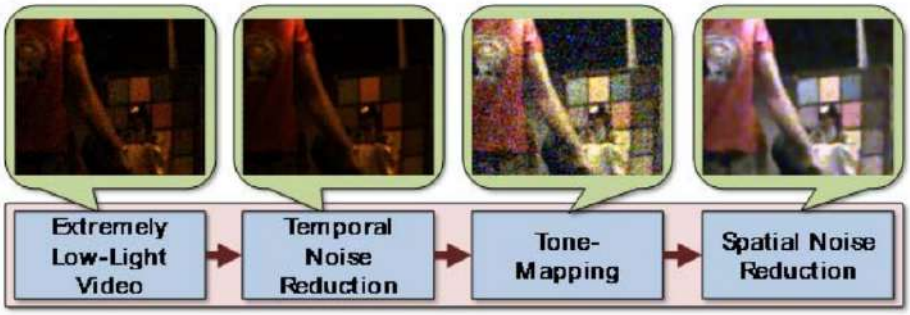

Figure 1.1: Early works in low-light video enhancement that attempts to model signal and noise without deep learning techniques.

However, recent data-driven attempts at enhancing algorithms using modern deep learning techniques implicitly learn these signal and noise models and give satisfactory results. A major challenge to training such algorithms is the lack of pixel-wise paired training datasets for low-light and normal light videos as it is impossible to collect real temporally consistent data under controlled settings with sufficient diversities and scale and noise of identical characteristics. Most of the current research in this field use single image methods that lead to results which are temporally inconsistent. Additionally, here the final attempt is to try and incorporate audio from the original videos as an addition to the feature space to see if that improves the performance of these models. Hence, the dataset required would comprise of 3 things, a low-light video with low SNR, a normal light video with high SNR and the corresponding audio files for these videos. Such a dataset as expected is not available currently.

1.3 Organization of Thesis

The outline of the thesis is as follows. The second chapter deals with the an elaborate description of the various methods explored to generate the dataset needed for this project. In chapter 3, the model for low-light video enhancement that does not use the audio files of the dataset is explored in more detail. Chapter 4 tries to explore adding audio features to this dataset as an additional feature space to see if any significant increase in output quality and SNR can be made. Finally, Chapter 5 concludes the thesis with the future scope of this project and further work that can be done. Appendix contains some additional results that were obtained from the models. Images and overall comparisons of performance are all done for the relevant models in the concerned chapters. References for this project have been cited at the end of the thesis.

CHAPTER 2

DATASET

Creating a dataset curated for this experiment is a very big challenge. We require 3 modes of data to perform the experiments proposed. Low-light videos, their corresponding normal light videos and the audio files. Another caveat is that the audio files must match the action being performed in the video. For instance, a video of a race car on track must have the audio of the race car's motion and not that of the commentator talking. This is because unless the audio has information relevant to the motion being tracked in the video, the chances of presenting any useful information is very minimal.

2.1 Existing datasets

The dataset that is relevant to this project is the one that is used for multi-modal research. Multi-modal research focuses on problems that involve both audio as well as video aspects of the data to generate insights, create new videos and several other tasks. Hence, they contain two of the required modes of input data. Here, we have used some of the videos from the following datasets.

Table 2.1: Existing datasets in LLIE and LLVE research

Name	Number	Format	Real/Syn	Video	Audio
LOL	500	RGB	Real	No	No
SICE	4413	RGB	Real	No	No
MIT-Adobe FiveK	5000	raw	Real	No	No
SID	5094	raw	Real	No	No
DRV	202	raw	Real	Yes	No
SMOID	179	raw	Real	Yes	No

The Audio-Visual Event (AVE) dataset Tian *et al.* (2018), that contains 4143 videos covering 28 event categories and videos. The dataset covers a wide range of audio-visual events (e.g., man speaking, woman speaking, dog barking, playing guitar, and frying food etc.) from different domains, e.g., human activities, animal activities, music

performances, and vehicle sounds. Each event category contains a minimum of 60 videos and a maximum of 188 videos, and 66.4% videos in the AVE contain audio-visual events that span over the full 10 seconds. Audioset Gemmeke *et al.* (2017) is another very large dataset created and maintained by the Machine Perception group at Google. It contains 632 audio event classes totalling 2,084,320 human-labeled 10-second sound clips drawn from YouTube videos.

Since creating a real time database like this is not possible, the next step is to go for synthetic creation of databases. Doing this would require to convert the normal light settings videos from the databases mentioned above into low-light settings.

2.2 Previous works

There are ways in which people have tried to convert normal setting videos into low-light. They are broadly in two categories, histogram based methods and data drive approaches.

2.2.1 Histogram based methods

There are previous works on converting low-light images to normal settings and vice-versa using methods like CLAHE (Contrast Limited Adaptive Histogram Equalization) and Histogram matching Banik *et al.* (2018). There are two commonly seen challenges with these methods that make them unsuitable for our task. Firstly, most of these methods work with black and white images. Significant results are not seen when applied to RGB color images. Secondly, works that deal with extending these methods to videos involve applying the same algorithm to each frame that fails to take into account any temporal consistencies in the final output.

2.2.2 Data driven approaches

This is a relatively new area and there are very limited approaches available here. One method proposed is shown in Triantafyllidou *et al.* (2020). The model proposed converts normal setting videos into low-light with a high degree accuracy and visual cor-

rectness.

Given the lack of available real short, long exposure video pairs the model uses a two-step approach that leverages data synthesis. The first step, involves training a dual CycleGAN model for the purposes of data synthesis. The dual CycleGAN maps video frames to a domain characterised by short exposure images. A domain bridge (e.g. long exposure images) is used to regularise the mapping with available paired supervision. The trained CycleGAN permits videos 'from the wild' to be projected into the long and then short exposure domains, thereby generating the necessary paired supervision. The second step, utilises this synthetic data to train a forward model capable of mapping low-light video RAW to long exposure RGB.

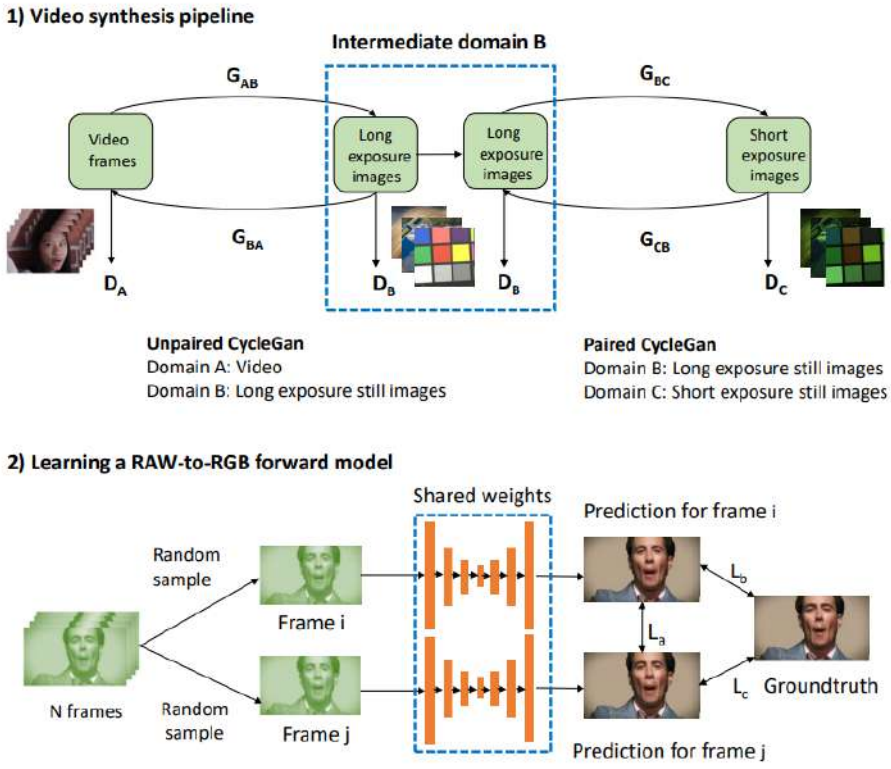

Figure 2.1: SIDGAN architecture that uses an intermittent domain to bridge the gap between the required input-output pair to generate synthetic low-light videos.

The proposed model SIDGAN, is modelled as a set of two CycleGANs in a dual configuration that learns the domain distributions for three domains; A, B and C. The model architecture is shown in Figure 2.1 and the domain B-C CycleGAN is shown in more detail in Figure 2.2. Domain A is characterised by a set of video frames defined by probability distribution $\{V_i\}_{i=1}^{N} \sim p_A$. The set of N videos from this domain are available for training. Similarly, they have considered M long exposure still images

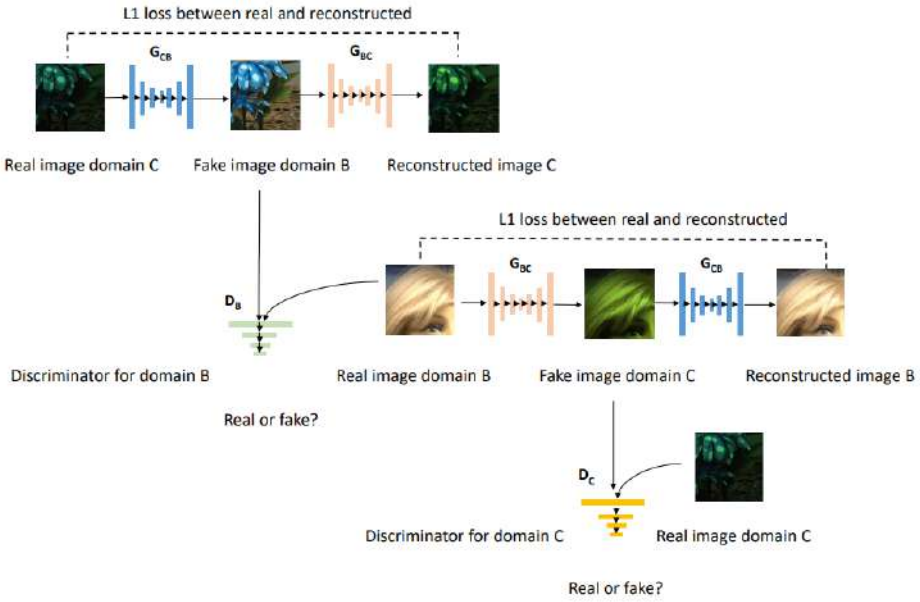

Figure 2.2: Visualisation of the Domain B-C CycleGAN, a sub-component of the complete dual CycleGAN architecture shown in Figure 2.1.

$\{L_i\}_{i=1}^{M} \sim p_B$ (domain B) and T short exposure still images $\{S_i\}_{i=1}^{T} \sim p_C$ (domain C).

2.2.3 Challenges with approach

While the above mentioned approach works very well for our purpose there are some challenges with using this method. The first is that the entire code for implementing this model is not currently made publicly available. Only excerpts and certain aspects are the code are available which makes it unsuitable to be directly implemented.

The second challenge is with training the model itself. They have employed the Vimeo-90K dataset Xue *et al.* (2019) to translate real-world videos into the low-light sensor specific domain. The dataset has 91, 701 septuplet samples, each containing 7 video frames of resolution 448×255. For the sensor-specific long and short exposure domains (i.e. domains B and C), they have used the Dark Raw Video (DRV) dataset Chen *et al.* (2019), which contains 224 low-light raw video data and corresponding long-exposure images. The intermediate domain B is represented by the long exposure DRV RGB images while for domain C, the provided preprocessed DRV short exposure RAW video frames were used. Since there are no pre-trained models available, re-writing and then training these models from scratch is computationally very expensive and not viable. Hence, this model could not be used directly for our purposes.

2.3 Proposed solution

As discussed earlier, low-light images differ from normal images due to two dominant features: low brightness/contrast and the presence of noise. In the proposed low-light synthesis pipeline, a transformation is fit onto the normal image to convert it into underexposed low-light image. The combination of linear and gamma transformation can approximate this job well.

2.3.1 Gamma correction

Gamma correction is a non-linear operation that is used to encode or decode luminance in image systems. It controls the overall brightness of the image. In the simplest cases, gamma correction is governed by the power-law expression

$$V_{out} = AV_{in}^{\gamma} \tag{2.1}$$

A gamma value $\gamma < 1$ is sometimes called an encoding gamma, and the process of encoding with this compressing power-law non-linearity is called gamma compression; conversely a gamma value $\gamma > 1$ is called a decoding gamma, and the application of the expansive power-law non-linearity is called gamma expansion.

2.3.2 Pipeline for images

The proposed pipeline for images consists of two steps: the first is a transformation that is applied on the image and the second is the noise function. This is shown clearly in Figure 2.3.

The low-light image transformation can be described as

$$I_{out}^{(i)} = 255 \times \beta \times \left(\frac{\alpha \times I_{in}^{(i)}}{255} \right)^{\gamma}, i \in \{R, G, B\} \tag{2.2}$$

where α and β are linear transformations, the I^{γ} represents the gamma transformation. The values of α, β and γ and how changes in their values can affect the final out-

come are discussed at a later section. The final values chosen are α from $U \sim [0.9, 1)$, β from $U \sim [0.5, 1)$ and γ from $U \sim [2, 3.5)$.

As for the noise, the model takes this into account by using the Gaussian-Poisson mixed noise model and simulate real low-light noise.

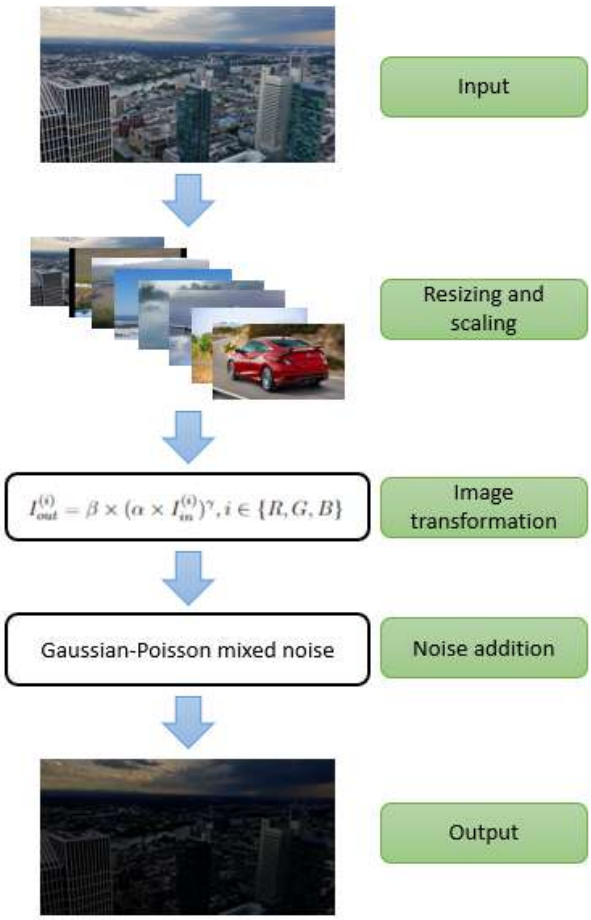

Figure 2.3: Pipeline used for generating low-light images.

2.3.3 Variation of α, β and γ

α, β and γ are the 3 parameters that control the low-light image transformation as described in Equation 2.2. Since the image as a 3 channel RGB image, all the three parameters are applied to the 3 channels without any change in value for a given pixel. The variations in the alpha parameter can be seen in Figure 2.5. The best visual fit for a low-light image appears at around $\alpha = 0.95$. Hence alpha is sampled from a uniform distribution within the limits [0.9, 1) to ensure diversity in output sample.

The variations for the β parameter can be seen in Figure 2.6. The parameter was

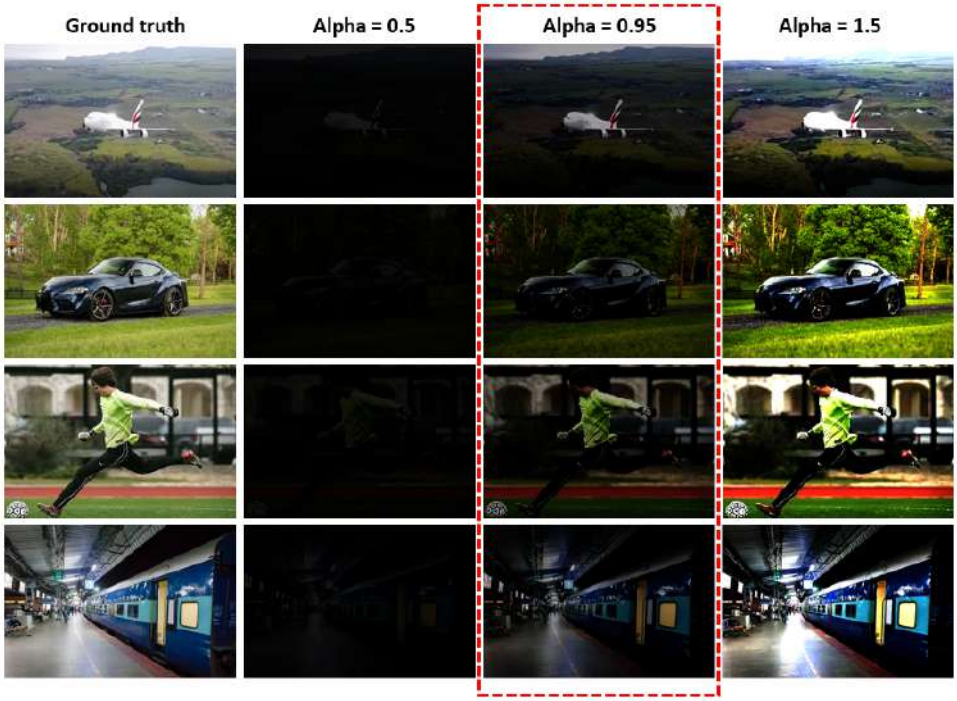

Figure 2.4: Variation of image quality with changes in α parameter keeping β at 0.75 and γ at 3. The best results are when α lies in the range of [0.9, 1).

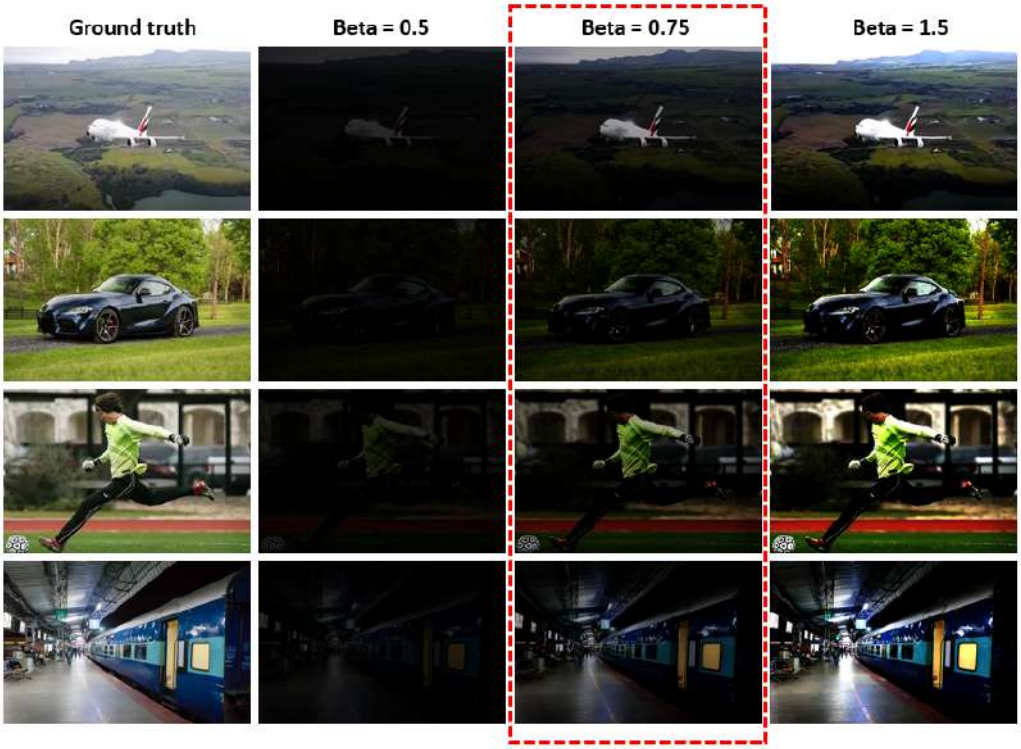

Figure 2.5: Variation of image quality with changes in β parameter keeping α at 0.95 and γ at 3. The best results are when β lies in the range of [0.5, 1).

varied across 3 values in 0.5, 0.75 and 1.5. Again, it is clear that the values around 0.75 give the best fit for a typical low-light image. Similar to the α parameter, β parameter was sampled from [0.5,1) to ensure diversity and scale in the output images.

Lastly, variations of the γ parameter can be seen in Figure 2.7. The parameter was varied across 3 values in 1,3 and 5. The best fit was obtained for values around 3. The γ parameter is varied between [2, 3.5) for the final output images generated.

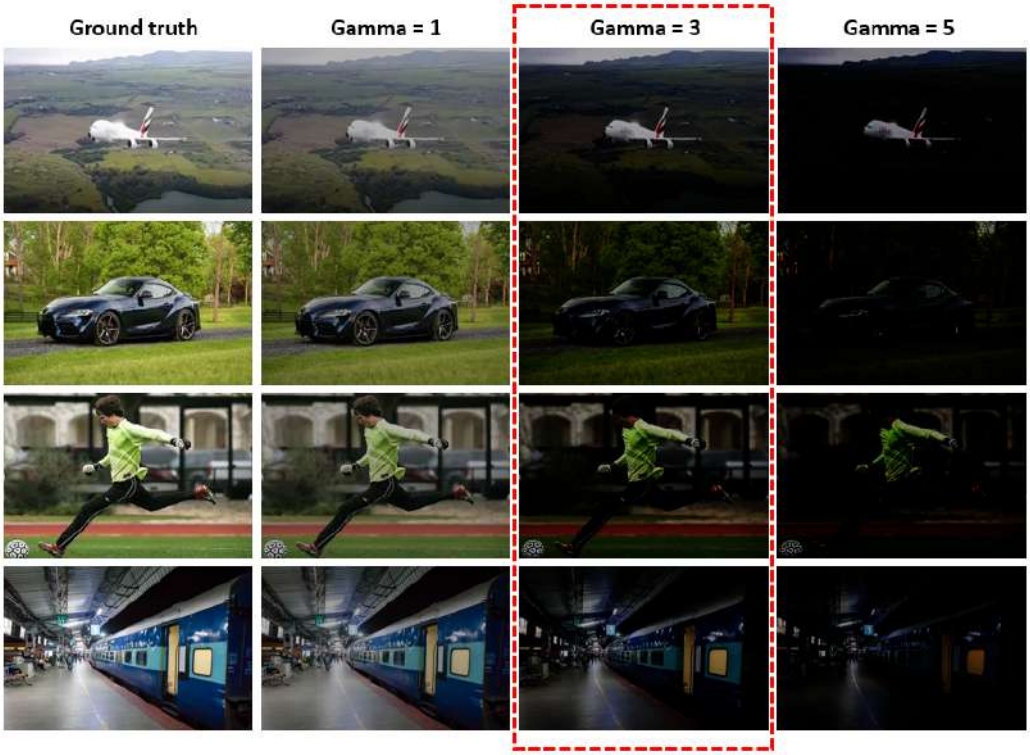

Figure 2.6: Variation of image quality with changes in γ parameter keeping α at 0.95 and β at 0.75. The best results are when γ lies in the range of [2, 3.5).

2.3.4 Temporal consistencies in videos

When the above model was applied directly to videos, frame by frame, there was a significant lack of temporal consistency. Lots of flickering was observed between the frames. In order to avoid this, a parameter sharing was proposed between the frames and applied pixel wise. It is governed by the equation

$$P_t = w_0 * P_t + w_1 * P_{t-1} + w_2 * P_{t-2} \tag{2.3}$$

where P_t refers to the parameters α, β and γ for frame t, $1 \leq t \leq N_{frames}$. Also, $w_0 + w_1 + w_2 = 1$. They were assigned as 0.5, 0.3 and 0.2 respectively.

2.3.5 Advantages of using this method for low-light conversion

There are some existing datasets for low-light image enhancement. However, these datasets still have their own limitations. In this section, the differences between the synthetic dataset and other low-light image enhancement datasets have been highlighted, to show that this synthetic dataset is a good complement to existing datasets.

Low-light ranges: Covering a large range of low-light conditions is an important factor for the generalization capabilities of the trained models. Given that the model parameters, namely alpha, beta and gamma are taken from a uniform distribution randomly, this generates sufficiently varying degrees of illumination and thus ensures that the models trained on this dataset generalize well to all degrees of illumination conditions.

Scale and diversity: Having a large dataset with diverse scenes and lighting conditions is significant for training a model that can generalize well. Manually collecting images or editing images as in other datasets is a costly and time consuming process, which make it hard to acquire data at scale. Therefore, existing datasets are all relatively small in size. In contrast, since this data generation is based on simulation, it can synthesize paired low-light and normal light images as much as needed for different scenes.

Compatibility Beside making the visual quality more appealing, improving the performance of other vision systems under low-light conditions is another important application for low-light enhancement. However, existing datasets do not contains manual annotations as they are only designed for visual quality enhancement. In contrast, the synthetic dataset can directly use existing public datasets to render low-light images and keep their corresponding annotations, such as bounding boxes for object detection and semantic segmentation masks. Thus, the synthetic dataset also has potential ability to improve the performance of fundamental vision methods to handle low-light conditions, such as object detection and semantic segmentation, etc.

Figure 2.7: Difference between converting without sharing any parameters and converting with sharing parameters. The red bound boxes clearly indicate the difference in quality seen.

Figure 2.8: Another example of the difference between converting without sharing any parameters and converting with sharing parameters.

CHAPTER 3

Basic video enhancement model without audio

This chapter of the report deals with the first part of video enhancement algorithms that were tried out. Before directly moving to using audio features, first a direct video enhancement using just the paired input, output features were tried out. The next chapter deals with extending this architecture to incorporating audio features as well.

There are many research works that explore the applications of deep convolutional neural networks for image-to-image restoration tasks (Shelhamer *et al.* (2017), Chen *et al.* (2017)). Recent developments in the field of U-Net architectures (Chen *et al.* (2018)) have shown very good results.

Based on this, the proposed architecture for this model was also taken to be a U-Net architecture. The U-Net video enhancement pipeline proposed by Jiang and Zheng (2019) was chosen as the basis on which the architecture was built.

3.1 Architecture used

The architecture initially proposed by Jiang and Zheng (2019) was used as the baseline for this model. However, this architecture was initially proposed for conversion of RAW sensor data in GRGB Bayer format into RGB color output as videos. Given, that here we already have RGB images as input, the initial layers of the model were accordingly modified to accommodate this change.

3.1.1 U-nets

Semantic segmentation, also known as pixel-based classification, is an important task in which we classify each pixel of an image as belonging to a particular class. U-net was originally invented and first used for biomedical image segmentation. Its architecture can be broadly thought of as an encoder network followed by a decoder network.

Unlike classification where the end result of the the deep network is the only important thing, semantic segmentation not only requires discrimination at pixel level but also a mechanism to project the discriminative features learnt at different stages of the encoder onto the pixel space.

The encoder is the first half in the architecture diagram. It usually is a pre-trained classification network like VGG/ResNet where you apply convolution blocks followed by a maxpool downsampling to encode the input image into feature representations at multiple different levels.

The decoder is the second half of the architecture. The goal is to semantically project the discriminative features (lower resolution) learnt by the encoder onto the pixel space (higher resolution) to get a dense classification. The decoder consists of upsampling and concatenation followed by regular convolution operations.

There were multiple reasons for adopting this particular architecture as the baseline. First and foremost was the ease with which a parallel audio processing encoder architecture could be added to a U-Net. More on this topic will be discussed in Chapter 4, when audio features spaces will be talked about. Secondly, low-light image enhancement algorithms directly applied to each frame of a video can easily cause flickering problems. To avoid such drawbacks and take advantage of temporal information, 3D convolution layers were adopted in this particular network to substitute for traditional 2D ones. All the layers were not directly upgraded to 3D convolutional layers. As shown in the paper, 2D pooling and deconvolution layers help networks perceive abstract elements in feature maps whereas 3D versions of them could mishandle sequential temporal information. Hence this particular modified U-Net combined 3D convolutions and 2D pooling/deconvolutions together to better integrate spatial-temporal information.

3.1.2 Model summary

Figure 3.1 illustrates the U-net architecture that was used. Like the standard u-net, it has an analysis and a synthesis path each with four resolution steps. In the analysis path, each layer contains $3 \times 3 \times 3$ convolutions each followed by a rectified linear unit (ReLu), and then a $2 \times 2 \times 2$ max pooling with strides of two in each dimension. In the synthesis path, each layer consists of an up-convolution of $2 \times 2 \times 2$ by strides of two in

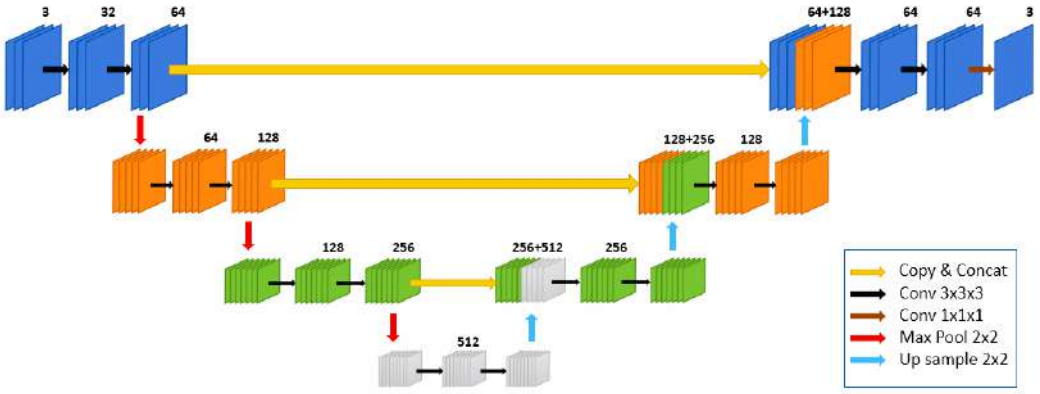

Figure 3.1: The standard 3D U-Net architecture as used in the model. The number of channels is mentioned above each block.

each dimension, followed by 3 × 3 × 3 convolutions each followed by a ReLu. Shortcut connections from layers of equal resolution in the analysis path provide the essential high-resolution features to the synthesis path. In the last layer a 1×1×1 convolution reduces the number of output channels to the number of labels which is 3 in our case.

3.2 Implementation

This model was trained on videos taken from the AVE dataset mentioned above. Since there are currently no constraints on the audio present in these videos and the content within the videos as well, this model was trained on 10 sec clips at 25fps. Learning rate was set to 10^{-4} initially and dropped to 10^{-5} after 15 epochs, 10^{-6} after 30 epochs. Training process proceeded for 70 epochs. The model used the L1 loss (see Equation 3.1).

$$L_1 loss = \sum_{i=1}^{H} \sum_{j=1}^{W} \sum_{k=1}^{C} \sum_{t=1}^{T} |y_{i,j,k,t} - y_{i,\hat{j},k,t}| \tag{3.1}$$

where H x W represents the dimensions of the image, C refers to the three R,G,B channels and T refers to the total number of frames within the given video clip.

The Adam optimizer was chosen for this particular problem. Adam (Kingma and Ba (2015)) is an adaptive learning rate optimization algorithm that's been designed specifically for training deep neural networks. Adam can be looked at as a combination of

RMSprop and Stochastic Gradient Descent with momentum. It uses the squared gradients to scale the learning rate like RMSprop and it takes advantage of momentum by using moving average of the gradient instead of gradient itself like SGD with momentum.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \left[\frac{\delta L}{\delta w_t} \right] \tag{3.2}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) \left[\frac{\delta L}{\delta w_t} \right]^2 \tag{3.3}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{3.4}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{3.5}$$

$$w_{t+1} = w_t - \hat{m}_t \left(\frac{\alpha_t}{\sqrt{\hat{v}_t} + \epsilon} \right) \tag{3.6}$$

where m_t represents aggregate of gradients at time t, w_t represents weights at time t, α_t represents learning rate at time t, represents derivative of loss function and δw_t represents derivative of weights at time t. β_1 and β_2 are generally constants fixed at 0.9 and 0.99.

3.2.1 Results

Figures 3.2 and 3.3 show the results of running this model on two low-light videos from the synthetic dataset that was generated. The first column shows the low-light video and the right side shows the normal setting video produced as output from the model. Visually, the model seems to do a fair job at enhancing the video. Metrics for comparing the outputs of this model using parameters like SNR (signal-to-noise ratio), SSIM (Structural Similarity) and MSE of MABD (Mean square error of Mean Absolute Brightness Differences) have been discussed in the next section.

Figure 3.2: Example of low light video frame and the converted normal light video

Figure 3.3: Another example of low light video frame and the converted normal light video

CHAPTER 4

LLVE with Audio features

There is currently no research that has been done on enhancing low-light videos using the audio features as information to augment the modelling. This is for varied reasons. The concept of low-light video enhancement as a topic has still not gained wide popularity. There are not many datasets available to do this kind of processing. The architectures that do perform low-light video enhancement either work with RAW sensor data Chen *et al.* (2019) or work with doing this operation frame by frame rather than taking temporal consistency into account Lv *et al.* (2018).

4.1 Dataset

Details about the dataset that has been used for this problem has been discussed at length in the first chapter. However, not all videos can be used directly for this application.

4.1.1 Assumptions

There are some assumptions that need to be kept in mind before attempting to include audio features in this kind of a problem.

Firstly, the audio itself needs to be related to motion that is happening within the video. For example, in a video that illustrates a car on a race track, audio about the car's motion as it passes by is acceptable but some other overlapping audio, for instance that of a commentator talking about the driver's speed or performance would not be acceptable. This constraint poses a major challenge on the nature and kind of videos that can be used for this experiment.

Secondly, the video itself should have only one central object whose motion we are tracking. A video of plane or a car or a person walking or kicking a ball is acceptable but if there are multiple objects within the video, then it would not be possible to track which object is causing the sound and which motion we should track.

4.2 Overall architecture

As mentioned earlier here, we have two kinds of inputs that need to be processed, the video files and also the corresponding audio inputs. Additionally, given the limited amount of training dataset available, the aim was try to utilise transfer learning from the previously trained U-Net architecture to better utilise the weights learned.

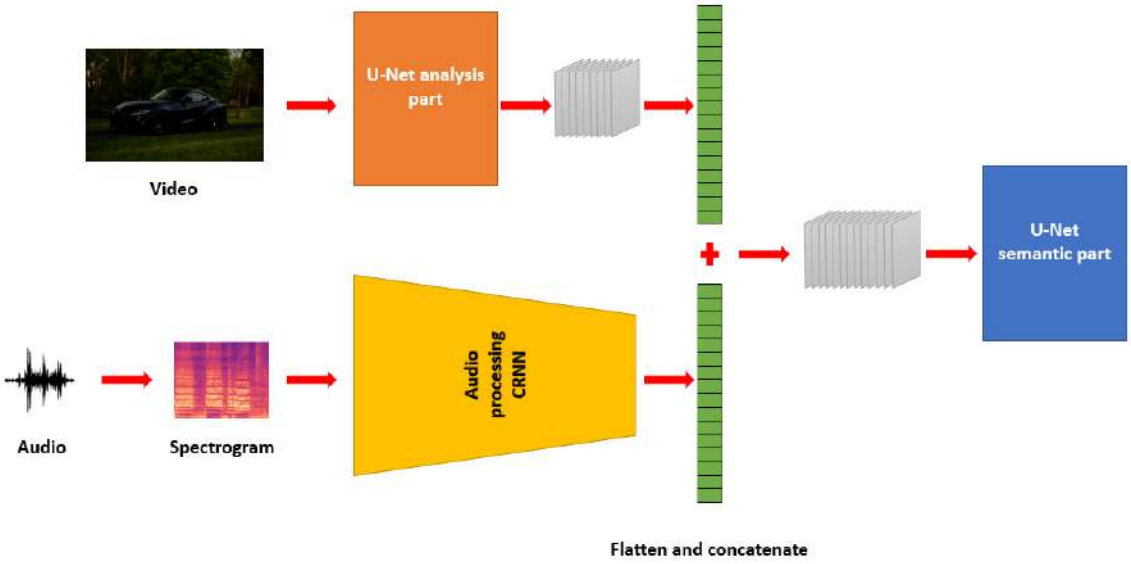

Figure 4.1: Overall multi-modal architecture for Low-light video enhancement using audio features as well

Figure 4.1 shows the overall architecture for how the system works. The U-Net architecture from the previous model is borrowed. The same trained final weights have also been used to introduce some amount of transfer learning into the model. The analysis part of the previous U-Net remains the same. After the last max pooling, the output feature map is flattened. Parallely, another branch converts the audio files into a Mel spectrogram (this is not part of the model and is actually done separately and the Mel spectrogram is directly fed into the model). The Mel spectrogram passes into a CRNN (Convolutional Recurrent Neural Network) explained later that is simply concatenated to the existing flattened U-Net. Rest of the architecture again remains the same as the U-Net with the filter sizes modified to accommodate the changes in the feature shape and size.

4.2.1 Spectrograms

Spectrograms are a means of using CNNs to work with audio data. A spectrogram of a signal plots its spectrum over time. It plots time on the x-axis and Frequency on the y-axis. It is as though, the spectrum is taken again and again at different instances in time, and then joined all together into a single plot. It uses different colors to indicate the amplitude or strength of each frequency. The brighter the color the higher the energy of the signal. Each vertical 'slice' of the spectrogram is essentially the spectrum of the signal at that instant in time and shows how the signal strength is distributed in every frequency found in the signal at that instant. Spectrograms are produced using Fourier Transforms to decompose any signal into its constituent frequencies.

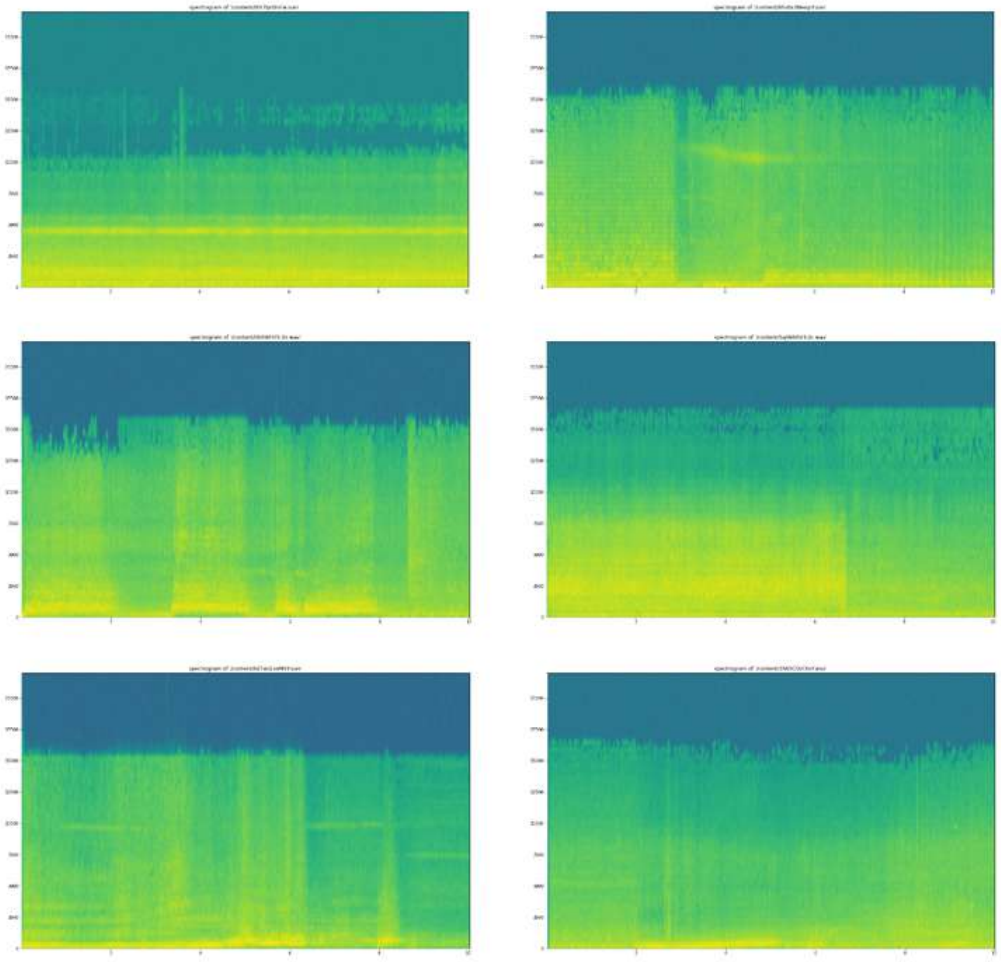

Figure 4.2: Samples of spectrograms of the audio signal that is present

Figure 4.2 shows the spectrogram of some of the signals present in the dataset that was collected. It is very clearly evident that no clear information can be found from

this image. The spectrograms all look very highly correlated and running these through any CNN model would not yield any conclusive results. Hence, there is a need for Mel spectrograms.

4.2.2 Mel spectrograms

Mel spectrograms are essential the same as the spectrograms but are viewed on the Mel scale. The Mel scale mimics how the human ear works, with research showing humans don't perceive frequencies on a linear scale. Humans are better at detecting differences at lower frequencies than at higher frequencies.

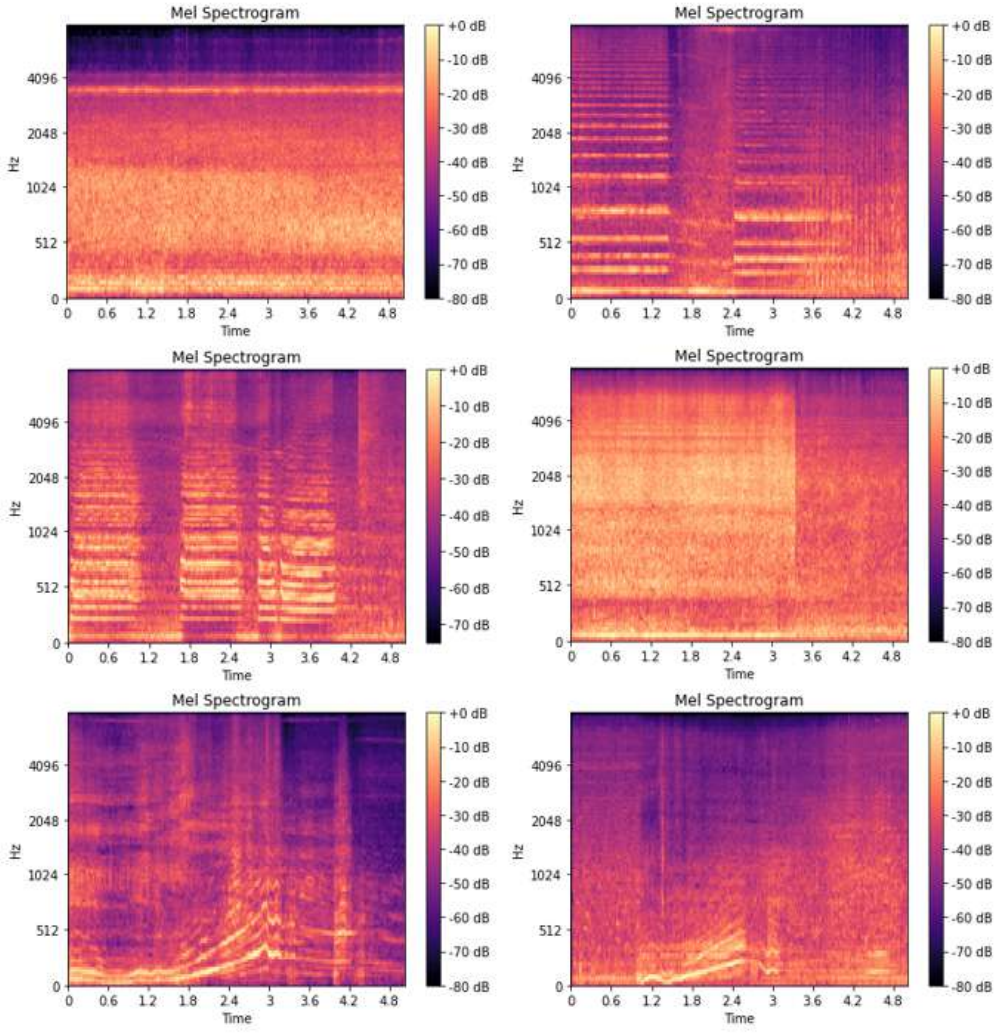

Figure 4.3: Corresponding Mel spectrograms of the audio signals shown in Figure 4.2. It is evident that a lot more differentiation and features can be seen.

Figure 4.3 shows the same samples as that seen in Figure 4.2 but this time, the spectrograms are computed on the Mel scale. The images now have a clear distinction and

feature maps with significant information can be generated from this images. Hence these were used as inputs into the CRNN model instead of directly using the spectrograms.

4.2.3 CRNNs

The Convolutional Recurrent Neural Networks is the combination of two of the most prominent neural networks: CNN(convolutional neural network) followed by the RNN (Recurrent neural networks). They have been shown to generate optimal results in audio signal processing tasks where temporal information is important Nasrabadi *et al.* (2014).

The architecture used here for audio signal processing has been borrowed from Nasrullah and Zhao (2019) with minimal changes to the pipeline except the last layer which has been flatten to combine with the U-Net semantic part. Figure 4.4 shows the pipeline for the same.

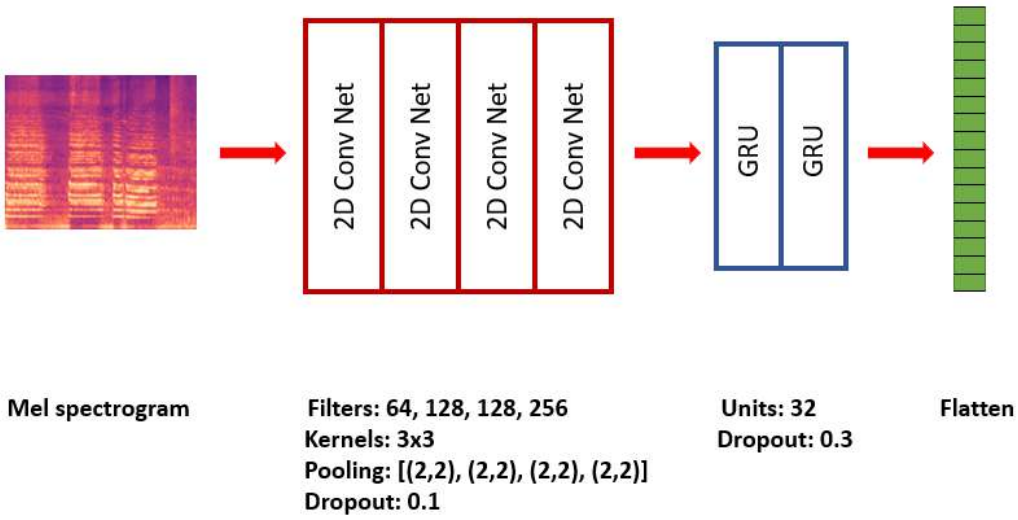

Figure 4.4: The audio processing part as shown in Nasrullah and Zhao (2019). Only modification is done to the last layer to flatten out the feature space.

The model has been trained on the synthetic dataset created as part of this project. It has 143 pairs of low-light and normal settings videos and their corresponding audio files which have been converted into Mel spectrograms as illustrated above. The same Adam optimizer and L_1 loss with same training parameters were used to ensure comparability between the two models.

4.3 Results

The results for the two runs can be seen in Figure 4.5 and Figure 4.6. Visually it is quite difficult to discern any differences between the two outputs. Hence, some performance metrics were used to evaluate these video based tasks.

4.3.1 Performance metrics

Three performance metrics were used to evaluate the results obtained on the models. The first two are PSNR (peak signal-to-noise ratio) and SSIM (Structural Similarity). They are commonly used metrics for evaluating performance on video based tasks. Additionally a third metric called MSE of MABD (Mean square error of Mean Absolute Brightness Differences) Jiang and Zheng (2019) was also used as it directly correlates to the task at hand

Peak signal-to-noise ratio (PSNR): The term peak signal-to-noise ratio (PSNR) is an expression for the ratio between the maximum possible value (power) of a signal and the power of distorting noise that affects the quality of its representation. Because many signals have a very wide dynamic range, (ratio between the largest and smallest possible values of a changeable quantity) the PSNR is usually expressed in terms of the logarithmic decibel scale. There are multiple ways of calculating PSNR (Nasrabadi *et al.* (2014)) for a video sequence as shown in this paper. Here, the first method of calculating the PSNR of individual frames is used. The final PSNR is the average of all individual frames.

$$PSNR_{vid} = \frac{1}{N} * 10 \sum_{i=1}^{N} log \frac{255^2}{MSE_i} \tag{4.1}$$

$$MSE_i = \sum_{i=1}^{H} \sum_{j=1}^{W} \sum_{k=1}^{3} (x_{i,j,k} - y_{i,j,k})^2 \tag{4.2}$$

Structural Similarity (SSIM): Structural Similarity Index compares images on 3 key metrics: luminance, contrast and structure. The comparison between the two images is performed on the basis of these 3 features. This system calculates the Structural Similarity Index between 2 given images which is a value between -1 and +1 where a

value of +1 indicates that the 2 given images are very similar or the same (often adjusted to [0,1] scale). Similar to PSNR, SSIM is calculated as average over all frames.

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{4.3}$$

where μ_x and μ_y are averages of x and y image intensities and σ_x^2 and σ_y^2 are corresponding variances. σxy is the covariance between x and y. $C_1 = (K_1L)^2$ and $C_2 = (K_2L)^2$ where L is the dynamic range (255 for RGB space) and $K_1 = 0.01$ and $K_2 = 0.03$.

Mean Absolute Brightness Differences (MABD): As described in the paper Jiang and Zheng (2019), MABD can be viewed as a general level of time derivatives of brightness value on each pixel location. It is calculated by the following equation for an RGB image.

$$MABD(t) = \frac{1}{3MN} \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{k=1}^{3} |br^{t+1}(i, j, k) - br^t(i, j, k)| \tag{4.4}$$

where (M,N) are the frame dimensions, $br^t(i, j, k)$ is the brightness of the pixel at location (i, j, k) at frame t with t being in the range of $1 \leq t \leq N_{frames}$.

The metric can be directly plotted as a vector over the frames or its MSE can be computed over all frames and used as a quantitative parameter to evaluate the outputs.

4.3.2 Frame wise comparisons

Given that there is very limited data to observe the results, two things were considered. One, the weights from the previous model were used as priors while training this model. Secondly, the model was evaluated using a stratified 6-fold approach as shown in Iwere and Rhee (2019). The entire dataset was split into test and train (120 videos and their audio files for training and the rest for testing).

Figure 4.5 and Figure 4.6 show the outputs obtained from the two models. It can be seen that there is a small improvement in the color and sharpness of images in some of the frames. But visually the improvement is not very clear.

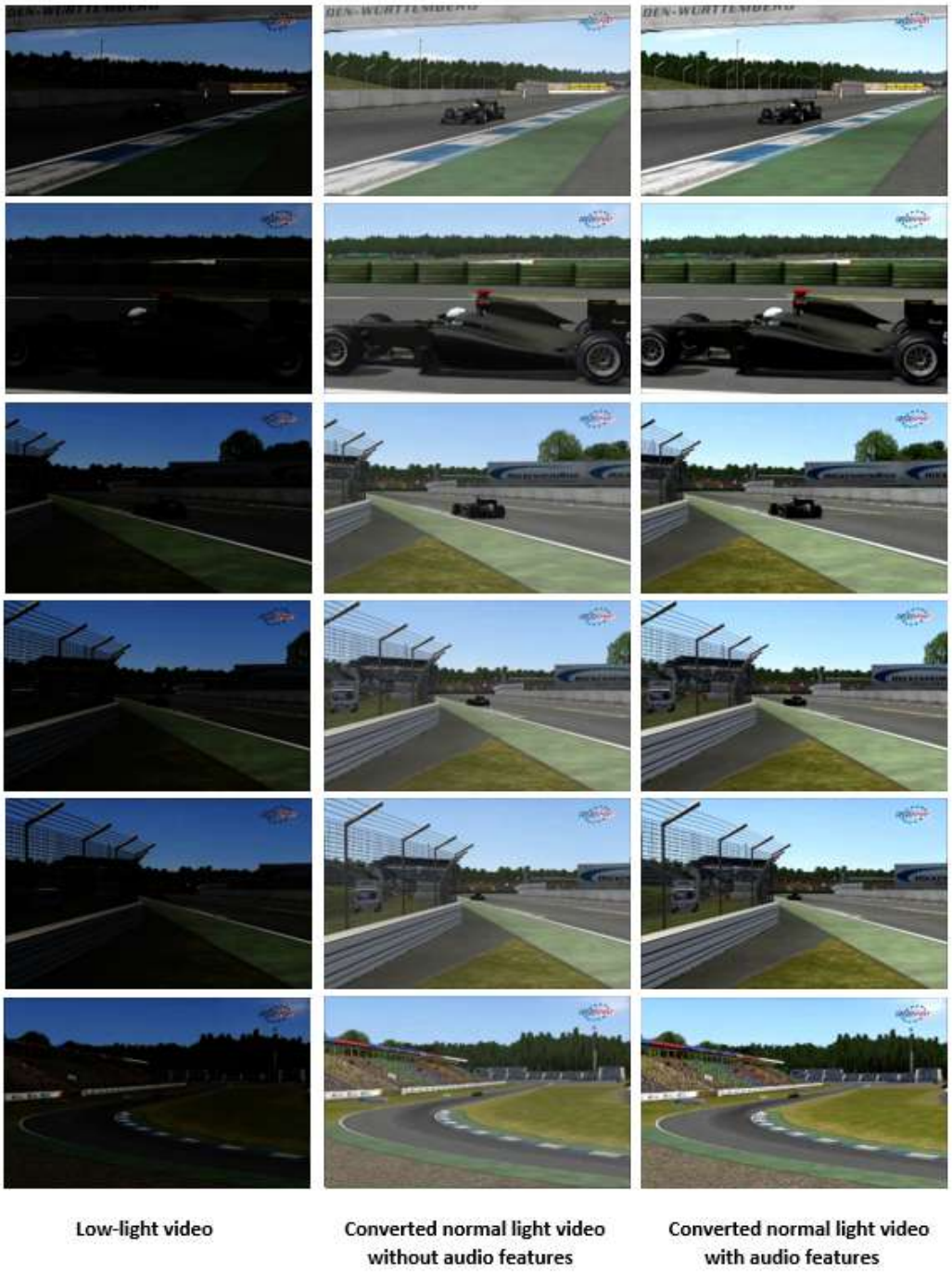

Figure 4.5: Results on the two models. First column shows the low-light video, the second column shows the converted video frames without using audio features and the third shows with audio features

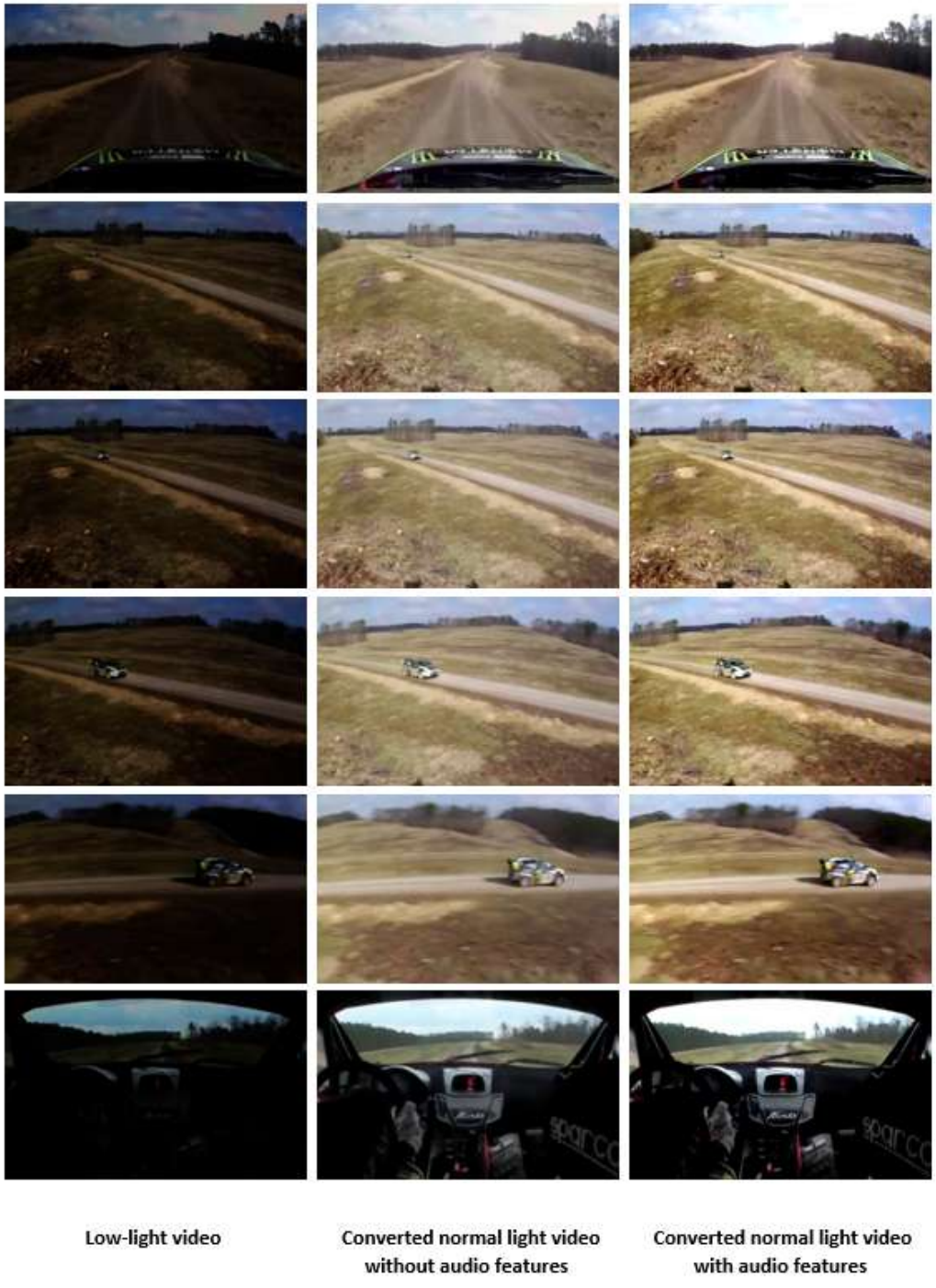

Figure 4.6: Another example of the results on the two models. First column shows the low-light video, the second column shows the converted video frames without using audio features and the third shows with audio features

Given that visually, the outcome of the two models is not very clear, some performance metrics mentioned above were evaluated. The Table 4.1 shows the final performance metrics as evaluated over the original LLVE U-Net architecture as well as the modified one using audio features. Individual metrics for each fold are shown included in the Appendix.

Table 4.1: Overall performance comparision between the two models

Metric Used	LLVE without Audio	LLVE with Audio
PSNR	22.16	**23.24**
SSIM	0.72	**0.73**
MSE(MABD)	**21.67**	23.28

As can be seen from the table above, there is a slight improvement in the PSNR and SSIM for LLVE with Audio features when compared to those without using the Audio features. However, the MSE(MABD) is better for the first case. The improvement, however, is very small. This can be attributed to two factors, first the lack of diversity in training data. (Refer appendix for information on event classes of synthetic data generated). Secondly, the architecture used to process the audio files is quite simple. More complex architectures using optical flow can be experimented with to generate greater increase in PSNR and SSIM.

CHAPTER 5

Conclusions and Future Work

Through this project, the aim has been to look at whether adding audio features to low-light video enhancement can improve the performance of these algorithms.

We have looked at a new way of generating synthetic low-light videos that takes care of the flickering problem by modifying existing low-light image synthesis pipelines that use gamma correction. This has been used to generate new videos for the project.

The first model is adapted from existing U-Net architecture that directly converts low-light RGB videos to normal light. This was used as the benchmark to compare with adding an additional branch in parallel that takes care of the audio input. The audio input was fed into the network by converting it into Mel spectrograms and using CRNNs to generate feature maps. There seems to be some improvement in the final output as a result of using audio features as well. However, the improvement is quite small due to constraints on the kind of videos that can be processes as well as the diversity present within the videos.

Future works could include generating and expanding over the dataset using the method proposed. More complicated architectures that utilise optical flows and/or more deeper CRNN architectures to analyze audio spectrograms can generate a greater increase in PSNR and SSIM. Future works could also include looking at dual and spatial audio which would contain more information. The network could also be modified for very specific applications including re-animating damaged video files, improving CCTV footages taken at night, etc.

APPENDIX A

Tables

Table A.1: Synthetic Data distribution

Event name	Number of videos
Car	46
Bus	22
Train	18
Plane	15
Bike	14
Ball	10
Kicking	10
Animal	8
Total	143

Table A.2: Results for 1st fold in the stratified 6-fold training

Metric Used	LLVE without Audio	LLVE with Audio
PSNR	22.35	**22.39**
SSIM	0.712	**0.721**
MSE(MABD)	**22.61**	24.18

Table A.3: Results for 2nd fold in the stratified 6-fold training

Metric Used	LLVE without Audio	LLVE with Audio
PSNR	24.14	**25.1**
SSIM	0.732	**0.74**
MSE(MABD)	**20.04**	21.24

Table A.4: Results for 3rd fold in the stratified 6-fold training

Metric Used	LLVE without Audio	LLVE with Audio
PSNR	21.26	**21.64**
SSIM	0.707	0.707
MSE(MABD)	**24.67**	25.32

Table A.5: Results for 4th fold in the stratified 6-fold training

Metric Used	LLVE without Audio	LLVE with Audio
PSNR	22.18	**22.34**
SSIM	0.721	**0.733**
MSE(MABD)	**21.67**	25.35

Table A.6: Results for 5th fold in the stratified 6-fold training

Metric Used	LLVE without Audio	LLVE with Audio
PSNR	19.69	**21.12**
SSIM	0.693	**0.706**
MSE(MABD)	**18.51**	20.91

Table A.7: Results for 6th fold in the stratified 6-fold training

Metric Used	LLVE without Audio	LLVE with Audio
PSNR	23.34	**26.85**
SSIM	0.755	**0.773**
MSE(MABD)	**22.52**	22.68

REFERENCES

1. **Banik, P. P.**, **R. Saha**, and **K.-D. Kim**, Contrast enhancement of low-light image using histogram equalization and illumination adjustment. *In 2018 International Conference on Electronics, Information, and Communication (ICEIC)*. 2018.

2. **Chen, C.**, **Q. Chen**, **M. Do**, and **V. Koltun**, Seeing motion in the dark. *In 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

3. **Chen, C.**, **Q. Chen**, **J. Xu**, and **V. Koltun** (2018). Learning to see in the dark. *CoRR*, **abs/1805.01934**. URL `http://arxiv.org/abs/1805.01934`.

4. **Chen, Q.**, **J. Xu**, and **V. Koltun** (2017). Fast image processing with fully-convolutional networks. *CoRR*, **abs/1709.00643**. URL `http://arxiv.org/abs/1709.00643`.

5. **Gemmeke, J. F.**, **D. P. W. Ellis**, **D. Freedman**, **A. Jansen**, **W. Lawrence**, **R. C. Moore**, **M. Plakal**, and **M. Ritter**, Audio set: An ontology and human-labeled dataset for audio events. *In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017.

6. **Jiang, H.** and **Y. Zheng**, Learning to see moving objects in the dark. *In 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

7. **Kingma, D. P.** and **J. Ba**, Adam: A method for stochastic optimization. *In* **Y. Bengio** and **Y. LeCun** (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. 2015. URL `http://arxiv.org/abs/1412.6980`.

8. **Lv, F.**, **F. Lu**, **J. Wu**, and **C. Lim**, Mbllen: Low-light image/video enhancement using cnns. 2018.

9. **Nasrabadi, A. T.**, **M. A. Shirsavar**, **A. Ebrahimi**, and **M. Ghanbari**, Investigating the psnr calculation methods for video sequences with source and channel distortions. *In 2014 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*. 2014.

10. **Nasrullah, Z.** and **Y. Zhao**, Musical artist classification with convolutional recurrent neural networks. *In 2019 International Joint Conference on Neural Networks (IJCNN)*. 2019.

11. **Shelhamer, E.**, **J. Long**, and **T. Darrell** (2017). Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **39**(4), 640–651.

12. **Tian, Y.**, **J. Shi**, **B. Li**, **Z. Duan**, and **C. Xu**, Audio-visual event localization in unconstrained videos. *In Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

13. **Triantafyllidou, D.**, **S. Moran**, **S. McDonagh**, **S. Parisot**, and **G. Slabaugh**, Low light video enhancement using synthetic data produced with an intermediate domain mapping. *In* **A. Vedaldi**, **H. Bischof**, **T. Brox**, and **J.-M. Frahm** (eds.), *Computer Vision – ECCV 2020*. Springer International Publishing, Cham, 2020. ISBN 978-3-030-58601-0.

14. **Xue, T.**, **B. Chen**, **J. Wu**, **D. Wei**, and **W. T. Freeman** (2019). Video enhancement with task-oriented flow. *International Journal of Computer Vision (IJCV)*, **127**(8), 1106–1125.

15. **Yiwere, M.** and **E. J. Rhee**, Sound source distance estimation using deep learning: An image classification approach. *In Sensors (Basel, Switzerland), 20(1), 172*. NLM, 2019.