Enhancement of Images Under Low Light Conditions Using Artificial Intelligence

Mazhara Marzook[1], Madhawa Herath[1], Migara Liyanage[1], Thakshila Thilakanayake[1]
[1]Department of Mechanical Engineering, Faculty of Engineering, Sri Lanka Institute of Information Technology, New Kandy Rd, Malabe, 10115, Sri Lanka
mazhara.marzook@gmail.com, madhawa.h@sliit.lk, migara.l@sliit.lk, thakshila.t@sliit.lk

ABSTRACT

Images taken in low light conditions do not contain all the information well-lit images contain. Various features including the colours of objects, details and the quality are lost. Extracting these features from images is very important for any kind of application of it. This study proposes a model to enhance the features of the image taken under low light conditions, by delivering a solution which improves the quality of the image through Artificial Intelligence. Through the proposed method, the clarity of the image is improved, making it closer to a well-lit image equivalent. Both Image Processing and Deep Learning based techniques are explored, including Convolutional Neural Network (CNN) based generative models. The Generative models considered are Autoencoders (AE) and Generative Adversarial Networks (GANs). The study has been carried out by using several datasets combined together, which include image pairs of well-lit and low light images. A comparison between the two CNN-based generative models is carried out. Through the study, it is quantitatively found, by the Structural Similarity Index and supported by the Peak Signal to Noise Ratio, that the proposed CNN-based Autoencoder model overrides the proposed CNN-based GAN model. This is further supported by qualitative observations of the image results. Both models, however, greatly enhance the low light images, bringing to light features that were not visible beforehand, and also provide results with good colour accuracy. Through this research study, the methods and solutions to enhance low light images have been addressed, as well as providing a comparison between two suitable models, Autoencoders and GANs. The proposed solution is able to address many of the limitations existing in the extent literature.

KEYWORDS: *Autoencoder, Comparative Analysis, Convolutional Neural Networks (CNN), Generative Adversarial Network (GAN), Image Enhancement*

1 INTRODUCTION

In today's technologically advanced world, there is a significant demand for image-based applications. The importance of constantly evolving and improving results from image-based applications cannot be overstated. Image enhancement, as well, is a highly studied topic, and improvements by various methods are rapidly occurring, from the long-studied Digital Image Processing (Maini and Aggarwal, 2010) to more rapidly developing Machine Learning methods (Li et al, 2020). However, images taken at night are of significantly lower quality than images taken during the day. Here, many features can be missed, resulting in less useful information being gleaned from these images. An enhanced image can serve both aesthetic purposes as well as be important for businesses and households. As an example, when considering security, CCTV cameras require a nighttime image from which features such as vehicle details and human features can be observed. A common method used in increasing the visibility of low-cost CCTV cameras at night is with Infrared (IR) technology, in which colour accuracy cannot be achieved to a reasonable level. Colour night vision technology, a less budget-friendly option, has allowed surrounding light, if available, to be used as assistance (Lorex, 2022). However, when there is no surrounding light, CCTV images cannot provide all the required information,

https://doi.org/10.54389/KRIH8788

which is a key challenge to safeguard the occupants of an apartment complex or to ensure a warehouse is secure. Similarly, for images taken by a phone camera at night, a dark image with low visibility of the people and buildings serves no purpose, compared to a well-lit image which can capture the liveliness of the scene. Motivated by the rapid developments in Artificial Intelligence and Machine Learning in Image-based applications, this study has been conducted to enhance images taken under low light conditions, by applying the powerful nature of Convolutional Neural Networks (CNNs). This study will be examining two CNN-based generative models to improve low lit images up to a well-lit, daytime level.

Image enhancement in full colour is an important objective of this project. Black and white enhanced images, while useful, cannot provide all the details that may be required or wanted by the consumer. While colour images can be obtained, it is even more important of obtain accurate colour information, as there is an uncertainty in the colours obtained by the current methods used, as these methods result in an image largely affected by the lighting and the types of illumination in the scene, the poor camera performance and losses that can occur (MacDonald, 2007).

Image enhancement methods that utilize Artificial Intelligence most commonly used CNNs. A study found an algorithm to enhance the quality of weak contrast images, using multi-layer CNN. Here, weak contrast images were generated using public datasets, along with images captured by the authors, and a neural network with a trapezoidal convolutional kernel was used (Wang and Hu, 2019). The study was able to provide an algorithm with a high Structural Similarity Index (SSIM) (Bakurov et al, 2022) and higher Peak Signal to Noise Ratio (PSNR) (Joshi et al, 2016) compared to previous studies. This study, while able to enhance low contrast images, did not generate a well-lit equivalent image. A subsequent study used a newly proposed RSCNN (Remote-Sensing Convolutional Neural Networks) model (Hu et al, 2021) to enhance remote sensing low light images, further improving on the multi-layer model proposed prior. This model, while it did enhance images, did not bring it up to a daytime, well-lit level, instead presenting a well-defined low light image. Another study using CNN-based methods for image enhancement utilized Conditional GANs for semantic segmentation (Wang et al, 2017), to obtain photorealistic images from semantic label maps. Their model tested a day-to-night conversion model and a night-to-day conversion model for semantic segmentation and concluded a day-to-night converter to produce better results. For the study to be conducted, a night-to-day (low light to well lit) conversion process is required. On the other hand, Autoencoders are most commonly used in denoising and image compression tasks. As an example, a study conducted on image noise reduction by the use of a denoising Autoencoder using four different models determined the best model was that which had the greatest number of hidden layers (Yasenko, 2020). Therefore, in addition to CNN-based GANs, CNN-based Autoencoders look promising in providing a solution to enhance nighttime images using Artificial Intelligence. This study will explore CNN-based generative methods for image enhancement, specifically in generating a well-lit image which is able to be presented as a daytime equivalent for a low light, nighttime image. Here, in addition to GAN models, Autoencoders will also be considered for low light image improvement.

To perform the above task, first a suitable data set was collected. This dataset was then preprocessed, and the models for both the CNN-based Autoencoder and GAN models were designed. Following training, testing and validating the models, the two generative models were compared against each other and against past studies by using performance indices such as the Structural Similarity Index Matrix (SSIM), Feature Similarity Index Matrix (FSIM), Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR), as well qualitatively by observing the image results. Through this, the most suitable model from the proposed Autoencoder and GAN model can be decided upon.

The size and variety of the dataset used for this project will greatly affect the result. Here, the dataset will include pre-existing sets and those taken by a camera. Instead of considering daytime and nighttime images, medium and low exposure images will be used, for reasons detailed in section 2.1. This is a limitation of the project, and it is assumed that these camera images are a suitable alternative for day and night image pairs.

The following paper will next detail the design methodology of the study. The results will also be presented, including the performance index results and test image samples, as well as a comparison between the generative models and a comparison against past studies.

2 DESIGN METHODOLOGY

This section will explain the selection of the dataset, preprocessing steps, design and development of the CNN-based Autoencoder and GAN neural networks, as well as the selection of the performance indices. Figure 1 illustrates the design methodology for this study. The two generative models were designed parallelly and used the same dataset and image dimensions for an unbiased comparison. Google Colab is used to run the Autoencoder and GAN codes, written in Python, as this environment is cloud based and allows access to computing resources such as GPU runtimes, which was used to carry out the task. A standard class GPU runtime is an Intel Xeon CPU @2.20 GHz, 13 GB RAM, Tesla K80 accelerator, and 12 GB GDDR5 VRAM.

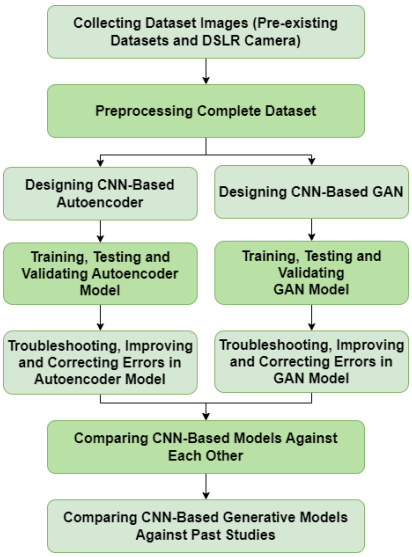


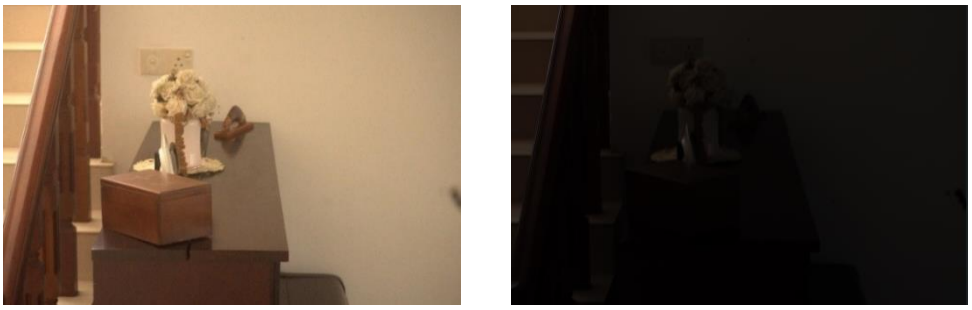
Figure 1. Illustration of Design Methodology

2.1 Dataset Collection and Selection

The requirement of the image dataset compiled is an image pair consisting of a well-lit image and its low light equivalent. It is essential that the two images are of the exact same scene. Therefore, obtaining a day and night image pair proves to be difficult, as many factors can vary as the time passes. To solve this issue, high and low exposure image pairs are considered as an alternative. Here, the low exposure image is low lit, and the high exposure image is well lit. Low exposure allows a lesser amount of light to enter the sensor of the camera, resulting in a darker image. This narrowed down the pre-existing image dataset options. Datasets such as Kaggle Day-Night (Mark, 2021), DeepISP (Schwartz et al, 2018), SID (Chen et al, 2018) and LOL (Wei et al, 2018) were considered. The image pairs obtained are as follows: 17 image pairs from Kaggle Day-Night Dataset, 110 image pairs from DeepISP Dataset, 33 image pairs captured by the author from a DSLR camera, 450 image pairs from LOL Dataset. Image samples are shown below, where Figure 2 shows a high/low exposure image pair taken from the LOL Dataset and Figure 3 shows a high/low exposure image pair taken from a DSLR camera.

(a) (b)

Figure 2. Sample Image Pair from LOL Dataset (Wei et al, 2018). (a) Well-Lit Image. (b) Low Light Image.



(a) (b)

Figure 3. Sample Image Pair taken from DSLR Camera (Author Captured). (a) Well-Lit Image. (b) Low Light Image.

When selecting the datasets, the Dark Zurich dataset was disregarded as the short exposure images, which were in *.raf* format, when converted resulted in an image with a purplish hue.

2.2 Preprocessing

The images selected must be resized to a lower quality, with the images brought to the same size. As the dataset contains high quality images taken from various DSLR cameras and phone cameras, they are of differing dimensions and formats, and cannot be directly passed to the models. Therefore, they are scaled down to a usable size of 200x134 pixels, as this is a size suitable for most of the images in the datasets. This size is reasonable for the Autoencoder and GAN models, which will require higher computation and power if higher quality images are used. The resizing is done by running an image processing script on Adobe Photoshop. Following the resizing, the images are cropped to 128x128, which is a square image. As the images will be undergoing various resizing processes, and the dimensions will be halved multiple times within the model, 128x128 (2^7x2^7) is deemed suitable. Initially, the 128x128 images were directly used, but it was found that resizing these cropped images up to 256x256 dimensions was able to yield better results, without a drastic increase in computation time. Following that, the images are then separated into train and test datasets, and the high exposure images are taken as the target image set while the low exposure images are the data which will be trained to reach target value. The train set consists of 550 image pairs, while the test set consists of 60 image pairs. These are saved as arrays. The shapes of these arrays are then checked, and then saved. In these image pairs, the low light image is the source image, and the well-lit image is the expected, or target, image.

2.3 Autoencoder Model

Autoencoders reconstruct the input by unsupervised learning and are commonly used in denoising and image compression (Bank et al, 2020). The input image is sent through hidden layers (the encoder) to obtain the compressed representation, and this is then sent through hidden layers to obtain the enhanced output image, via the decoder (Cunningham et al, 2020). The Autoencoder Model proposed is depicted in Figure 4.

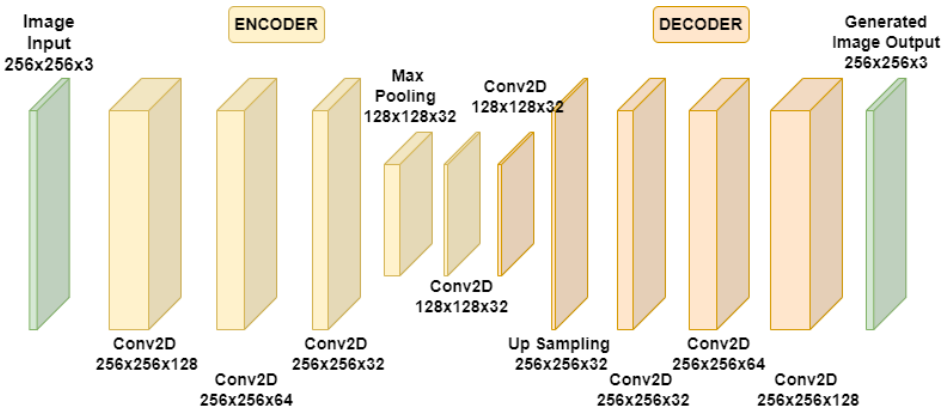


Figure 4. Proposed Autoencoder Model

The architecture for the proposed Autoencoder is shown below in Table 1.

Table 1. Autoencoder Architecture

Layer Type	Description	Output Shape	Activation Function
Image Input	Colored image input	256x256x3	
1st Encoder Hidden Layer (Convolutional)	128 filters, 3x3 kernel size	256x256x128	ReLU
2nd Encoder Hidden Layer (Convolutional)	64 filters, 3x3 kernel size	256x256x64	ReLU
3rd Encoder Hidden Layer (Convolutional)	32 filters, 3x3 kernel size	256x256x32	ReLU
Max Pooling Layer	2x2 max pooling	128x128x32	
4th Encoder Hidden Layer (Convolutional)	16 filters, 3x3 kernel size	128x128x16	ReLU
1st Decoder Hidden Layer (Convolutional)	16 filters, 3x3 kernel size	128x128x16	ReLU
Up Sampling Layer	2x2 up sampling	256x256x16	
2nd Decoder Hidden Layer (Convolutional)	32 filters, 3x3 kernel size	256x256x 32	ReLU
3rd Decoder Hidden Layer (Convolutional)	64 filters, 3x3 kernel size	256x256x64	ReLU
4th Decoder Hidden Layer (Convolutional)	128 filters, 3x3 kernel size	256x256x128	ReLU
Image Output	Colored generated image output	256x256x3	

Many types of layers were considered for the model, but the best results were obtained with the above structure. Here, the Convolutional layer is used along with the ReLU activation, as well as Max Pooling and Up Sampling layers. The convolutional layer can be considered as the backbone of CNN Models. It convolves data and passes on the transformed version to the next layer. ReLU Activation is used in order to prevent the computations from having exponential growth. For this cause, negative values of the matrix are substituted by 0 and the positive values are left as it is. In Max Pooling, for the selected filter size, the largest value in each patch of the feature map of the image is calculated and the image is then down sampled, reducing the dimensions of the image and highlighting the most important characteristics of the image, disregarding the less important factors. Up Sampling, does the opposite, increasing the image dimensions and bringing it back to the original size.

The above Autoencoder model is compiled by using an Adam optimizer with a learning rate of 0.001 and a loss function of Mean Square Error (MSE). Adam is used as the optimizer, as it is considered to be the best and fastest optimizer. For machine learning problems where a continuous output is obtained, such as the one in this study, MSE is the most suitable loss function. This is as it gives us the average of the squared difference between the generated and expected values for the images. The Autoencoder is then trained with a batch size of 50 for 50 epochs, and the results are tested, and the performance index results (SSIM, FSIM, MSE and PSNR) can be obtained.

2.4 Generative Adversarial Network Model

Generative Adversarial Networks have two neural networks competing to generate artificial data that can be taken as real data. A noise input to the generator is along with selected images from the dataset are sent to the discriminator, which tries to differentiate between real and fake images. Through this process, the generated images are finetuned by the model by changing its hyperparameters, until the discriminator is tricked (Hermosilla et al, 2021). The operation of a GAN model is illustrated in Figure 5.

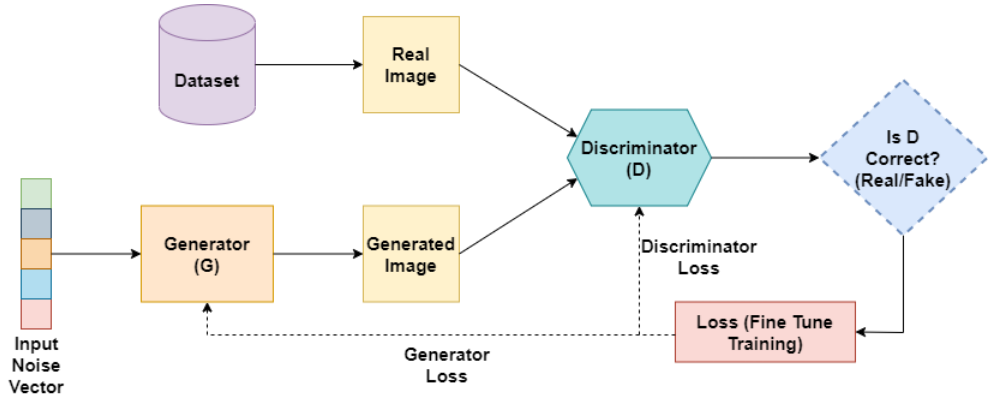


Figure 5. Basic GAN Model

To enhance low light images, instead of a noise input to the generator, the input is instead the low light image equivalents of the real image from the dataset. It is important to note that the images are normalized before they are passed through the training model, resulting in values between -1 and 1. The input to the generator, and thereby the GAN model, is an image of size 256x256x3. The output of the GAN is the output of the discriminator. The output layer of the generator is the same shape as the input layer of the discriminator, which is an image.

The generator model works towards creating a well lit image from the low light image. The generator consists of a model similar to the autoencoder model. Here, the only difference is that an activation function of tanh is used. This is used as the images have been normalized, bringing them to a value between -1 and 1. The tanh activation is the most suitable for this type of image, as it too gives an output between -1 and 1. However, the discriminator takes in two images, the real and fake. Next, real samples are generated, which are the real images from the dataset. These are given a value of 1, indicating that they are real. Following that fake images are generated, which are the predicted images by the generator model. The discriminator model gives an output which determines the realness or fakeness of the image. The Discriminator Model for the GAN proposed is shown in Figure 6.

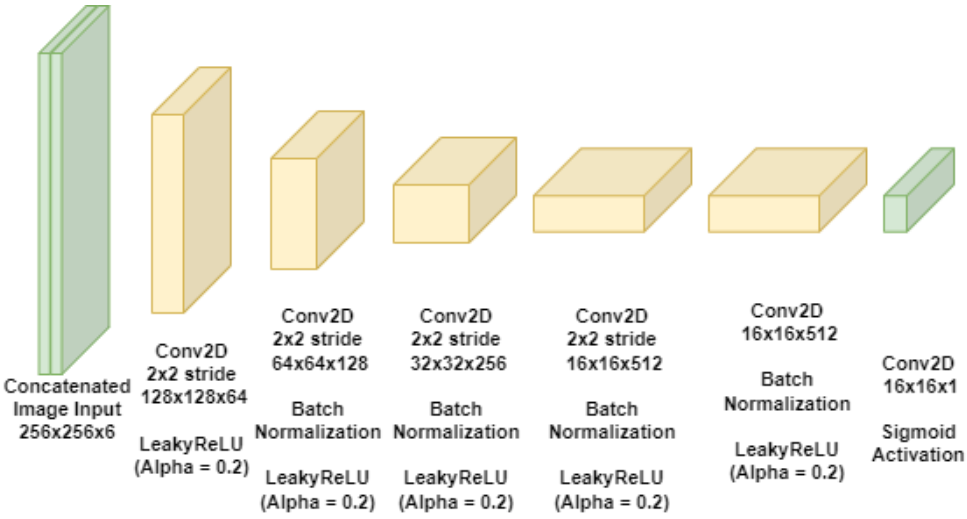


Figure 6. Proposed GAN Model

The architecture for the proposed discriminator is shown below in Table 2.

Table 2. GAN Discriminator Architecture

Layer Type	Description	Output Shape
Merged Image Input	Concatenation of generated image and dataset image (both colored)	256x256x6
1st Hidden Layer (Convolutional)	64 filters, 4x4 kernel size, 2x2 stride	128x128x64
Leaky ReLU Activation Layer	Alpha = 0.2	128x128x64
2nd Hidden Layer (Convolutional)	128 filters, 4x4 kernel size, 2x2 stride	64x64x128
Batch Normalization Layer		64x64x128
Leaky ReLU Activation Layer	Alpha = 0.2	64x64x128
3rd Hidden Layer (Convolutional)	256 filters, 4x4 kernel size, 2x2 stride	32x32x256
Batch Normalization Layer		32x32x256
Leaky ReLU Activation Layer	Alpha = 0.2	32x32x256
4th Hidden Layer (Convolutional)	512 filters, 4x4 kernel size, 2x2 stride	16x16x512
Batch Normalization Layer		16x16x512
Leaky ReLU Activation Layer	Alpha = 0.2	16x16x512
5th Hidden Layer (Convolutional)	512 filters, 4x4 kernel size	16x16x512
Batch Normalization Layer		16x16x512
Leaky ReLU Activation Layer	Alpha = 0.2	16x16x512
6th Hidden Layer (Convolutional)	1 filter, 4x4 kernel size	16x16x1
Sigmoid Activation Layer		16x16x1

In addition to the layers considered for the autoencoder above, the discriminator consists of LeakyReLU as the activation function. Batch Normalization was also used prior to feeding data into the neural network. LeakyReLU is a modified ReLU activation function, but it also plots some values close to zero to be negative, unlike ReLU. Here, alpha is the parameter in the LeakyReLU layer which represents that small negative value. Batch Normalization helps to stabilize the model and make it faster. A sigmoid activation layer is most suitable when the output required is a probability and is used as a value between 0-1 is needed as the output, as a probability of the "realness" of the output image. The above discriminator model is then compiled using the Adam optimizer with a learning rate of 0.0002.

Following that, the GAN model needs to be created, which is resulted by combining the generator and discriminator model. The GAN is then trained, with a batch size of 50 for 50 epochs. For the training process, first only the discriminator is trained, freezing the generator. Through this, the first set of discriminator weights are obtained. Next the generator is trained, keeping the discriminator frozen, and the above discriminator weights are used. Through this, the generator weights are also obtained. This process is repeated for the 50 epochs.

2.5 Comparison of Generative Models

To compare the two models against each other, Structural Similarity Index Matrix (SSIM) (Bakurov et al, 2022), Feature Similarity Index Matrix (FSIM) (Zhang et al, 2011), Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR) (Joshi et al, 2016) will be used.

These values are expected to be used to identify the success of the study, along with a qualitative comparison of the well-lit images to the generated output images.

The SSIM compares the original, clean reference image x with the changed or corrupt image y (Bakurov et al, 2022). The index is calculated as:

$$SSIM(x,y) = [l(x,y)]^{\alpha}.[c(x,y)]^{\beta}.[s(x,y)]^{\gamma} \quad (1)$$

where α, β and γ are weighing exponents. Here, $l(x,y)$ is the luminance comparison, $c(x,y)$ is the contrast-based comparison and $s(x,y)$ is the structural comparison. They are calculated as:

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{{\mu_x}^2 + {\mu_y}^2 + C_1} \quad (2)$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{{\sigma_x}^2 + {\sigma_y}^2 + C_2} \quad (3)$$

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \quad (4)$$

where σ is the standard deviation, which represents contrast and μ is the patch average for each image, which is a representation of the luminance. Here, C_1, $C_2$and C_3 are small constants introduced for numerical stability, such that they are a function of the range of pixel values L and the constants K_1 and K_2, which are usually $1x10^{-2}$ and $3x10^{-2}$. These values are calculated as:

$$C_1 = (K_1L)^2 \quad (5)$$

$$C_2 = (K_2L)^2 \quad (6)$$

$$C_3 = \frac{C_2}{2} \quad (7)$$

The SSIM result produces a value between 0-1, and a higher value indicates a higher similarity between the images.

PSNR uses the MSE to find the similarity between the two images (Joshi et al, 2016). This is given in dB, and the equation is as follows.

$$PSNR = 10log_{10}\left[\frac{I^2}{MSE}\right] \quad (8)$$

$$MSE = \frac{1}{[N \times M]^2}\sum_{i=0}^{N=1}\sum_{j=0}^{M=1}(x_{ij} - y_{ij})^2 \quad (9)$$

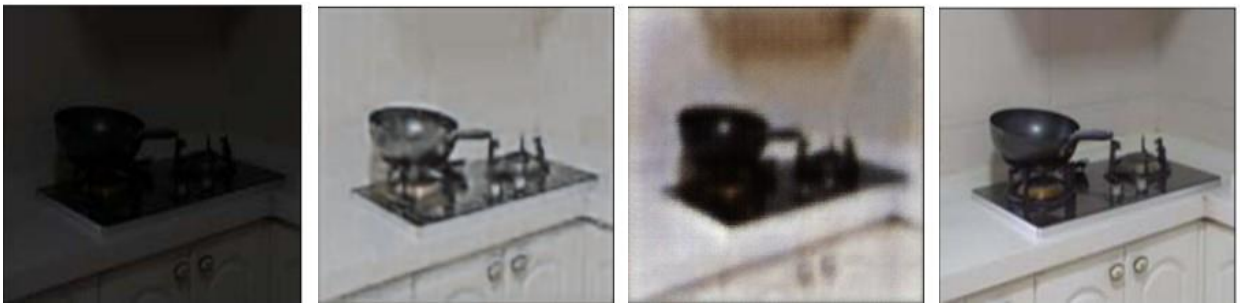
Here, I represents the maximum intensity of the grayscale image (most commonly 2^8-1), x is the original image and y is the changed image. x_{ij} represents the intensity of the ij^{th} pixel of the original grayscale image and y_{ij} is the same for the generated image. N and M are the number of rows and columns in the images x and y.

FSIM (Zhang et al, 2011) is used to support the results obtained from the above two methods. A high FSIM value indicates a higher similarity between the two images.

3 RESULTS

The results below are obtained after finetuning the Autoencoder and GAN models, as detailed Section 2.3 and 2.4. Here, the low light images from the test dataset are passed through the models, and the generated images can be obtained. For the GAN model, the test data is passed to the generator instead, to obtain an image result. The below Figures show a comparison between the source image— which is the low light image— to the Autoencoder generated image, GAN generated result, and finally the target, or expected, brightly lit image.

Figure 7 shows a scene depicting a pantry countertop, with the focus on the pan on the stove. As we can see, the generated images 7(b) and 7(c) are much brighter than the source, 7(a). The pan can easily be identified, although it is noted that the Autoencoder produces the pan in a lighter colour than the expected result, while the GAN results in a darker image. It can also be pointed out that the GAN result is not as clear as expected.



(a) (b) (c) (d)

Figure 7. Image 1 Comparison Between Models. (a) Low Light Image (Source Image). (b) Autoencoder Generated Image. (c) GAN Generated Image. (d) Well Lit Image (Expected Image).

Figure 8 consists of various textures, including a banket and soft toys. As we can see, the brightness as well as the visibility and the differentiation between the various textures has increased in the generated image, as compared to the low light image. The Autoencoder result easily distinguishes between the individual toys, while the GAN produces a more vivid, albeit unfocused, result.

(a) (b) (c) (d)

Figure 8. Image 2 Comparison Between Models. (a) Low Light Image (Source Image). (b) Autoencoder Generated Image. (c) GAN Generated Image. (d) Well Lit Image (Expected Image).

Figure 9 depicts a cupboard with various kitchen utensils and crockery. Here, the GAN produces images that are much darker than the Autoencoder model, but the colours allow the objects to be easily identified in both results.

(a) (b) (c) (d)

Figure 9. Image 2 Comparison Between Models. (a) Low Light Image (Source Image). (b) Autoencoder Generated Image. (c) GAN Generated Image. (d) Well Lit Image (Expected Image).

As shown above, generated images by both the Autoencoder and GAN enhance the images, by brightening the scenes. The objects in the images can be seen clearly without straining. The colors produced by the two models are adequately accurate as well. The Autoencoder model produced a sharper, more distinguishable image, compared to the more blurred image produced by the GAN model. The Autoencoder model, while it produces images that are more visually enhanced than the GAN, results in images that are slightly washed out compared to the GAN model.

The SSIM, FSIM, MSE and PSNR results for the two generative models for the first ten test data images are shown below in Table 3. Here, the original image is compared to the generated image.

Table 3. Performance Index Results for Generative Models

Image	SSIM		FSIM		MSE		PSNR	
	Autoencoder	**GAN**	**Autoencoder**	**GAN**	**Autoencoder**	**GAN**	**Autoencoder**	**GAN**
1	0.89	0.85	0.67	0.64	2.14	42.58	31.94	31.72
2	0.89	0.84	0.66	0.63	2.56	2.58	32.04	32.21
3	0.91	0.84	0.68	0.63	32.04	39.25	32.28	31.98
4	0.9	0.84	0.65	0.63	2.09	41.02	31.9	31.9
5	0.89	0.84	0.63	0.63	0.74	4.09	32.07	32.14
6	0.88	0.82	0.64	0.6	0.29	4.51	31.69	32.3
7	0.88	0.84	0.6	0.63	8.56	10.17	32.64	32.7
8	0.89	0.8	0.61	0.57	6.69	2.06	32.85	32.17
9	0.88	0.86	0.67	0.65	0.83	35.17	31.94	32.25
10	0.91	0.83	0.68	0.64	12.73	3.63	32.29	31.99

The SSIM gives the image quality degradation caused by various processes such as compression and losses due to data transmission. This value should be between 0 and 1, with a higher value indicating a closeness to the original (here, expected) image. Here, the results for the Autoencoder model are all above 0.88, with the highest results reaching above 0.9. For the GAN model, the results are all above 0.8, with the highest at 0.86. Therefore, it is noted that the Autoencoder model produces comparatively better results in terms of the Structural Similarity Index Matrix. This is most likely due to the blurriness in the images produced by the GAN model. FSIM is similar to SSIM, but instead focuses on features of the image. The Autoencoder values vary from 0.6 to 0.68 in these sample test images, while the results for the GAN model vary from 0.57 to 0.65. In terms of Feature Similarity Index Matrix as well, the Autoencoder model shows better performance results. Here, this occurs due to the features not being as distinguishable, due to the less sharp nature of the GAN model results.

The MSE gives the average of the square of the difference between the two images. Here, it is noted that there is a variation depending on the images, with some giving a small value of less than 1, and other images reaching comparatively high errors above 30. Overall, here too it can be noted that the Autoencoder model proposed results in lower errors in terms of Mean Square Error. Here, it is observed that the pixel wise difference can be large in some images for the models, and much less for others. As an example, in Figure 9, the inside of the cupboard, which is expected to be a white, is instead black in the GAN generated result. The MSE in the Autoencoder model can also be accounted for by the less vividness of the colours it produces.

The PSNR gives a ratio of the quality between the original and generated images, and therefore a higher value is preferable. Here, the PSNRs are all around 32dB, for both the generative models, producing extremely close results. The PSNR value is dependent on the MSE value and shows similar results.

A summary of the performance index results on the complete dataset of all test images is shown below in Table 4. The SSIM and FSIM values are converted to percentages, to show the variation more clearly. Higher SSIM, FSIM and PSNR values as well as lower MSE values are favorable.

Table 4. Performance Index Comparison of Generative Models

Performance Index	Proposed Autoencoder Model	Proposed GAN Model
SSIM	88.20%	83.72%
FSIM	65.73%	62.67%
MSE	14.65	17.70
PSNR	32.31 Db	32.35 Db

Here, there is a 4.5% difference in the SSIM results of the two models, with the Autoencoder showing a higher value. By considering the FSIM as well, the Autoencoder provides a 3% higher value. The Mean Square Error value is higher for the GAN model, showing that there is a larger error produced with the CNN-based GAN model proposed. However, when it comes to the PSNR, the GAN model shows a slightly better value. This 0.04 Db difference though, is insignificant compared to the rest of the values. Therefore, quantitatively, thE Autoencoder model provides better results than the GAN model.

The below table shows a comparison of the pSNR and SSIM values between past studies, which are "Learning to See in the Dark" (Chen et al, 2018), "RSCNN: A CNN-Based Method to Enhance Low-Light Remote-Sensing Images" (Hu et al, 2021), "Trying to See Low Exposure Images using CNN" (Shinde et al, 2021), and the designed Autoencoder and GAN model. Higher PSNR and SSIM values are favorable. The Learning to See in the Dark model was tested on two datasets.

Table 5. Comparison Between Generative Models and Past Studies

Model	PSNR (dB)	SSIM
Designed Autoencoder Model	32.31	0.8820
Designed GAN Model	32.35	0.8372
Learning to See in the Dark (Sony Dataset)	28.88	0.787
Learning to See in the Dark (Fuji Dataset)	26.61	0.680
RSCNN	28.194	0.825
Trying to See Low Exposure Images using CNN (CNN)	18.2029	0.8143

Therefore, it can be seen that the Autoencoder and GAN models developed within the study provide better results compared to the considered studies, with the designed Autoencoder Model producing the most favorable results.

4 CONCLUSION

Images taken under low lighting conditions are often not able to provide all the information needed, especially when related to security camera images. The purpose of this study is to enhance low light images, to result in images from which useful information can be gleaned. To carry out this task, Artificial Intelligence in the form of Convolutional Neural Network (CNN)-based generative models have been adopted. Here, by collecting a sizable dataset and considering various architectures for the models of these two generative approaches, a suitable architecture was obtained for both an Autoencoder and GAN. By performing a quantitative and qualitative comparison between the results obtained by both the Autoencoder and the GAN model, via observing the generated images visually and also by analyzing the results through performance indices such as SSIM, FSIM, MSE and PSNR along with other factors, the more suitable model for this task was found to be the CNN-based Autoencoder model. Both models, however, were able to greatly enhance the low light images. Here, the Autoencoder model was able to produce an average SSIM (Structural Similarity Index Matrix) result of 88.20%, compared to the average 83.72% obtained by the GAN model. This study into image enhancement using CNN-based generative models will therefore aid in future research in extracting information from low light images.

REFERENCES

Maini, R., Aggarwal, H. (2010). A Comprehensive Review of Image Enhancement Techniques. *Journal of Computing, 2*(3). https://arxiv.org/abs/1003.4053

Li, G., Yang, Y., Qu, X., Cao, D, Li, K. (2020). A Deep Learning Based Image Enhancement Approach for Autonomous Driving at Night. *Knowledge-Based Systems, 213.* https://doi.org/10.1016/j.knosys.2020.106617

Lorex. (2022). Color Night Vision https://www.lorex.com/pages/color-night-vision#:~:text=The%20camera%20sensor%20%20s%20ees%20this,lighting%20conditions%20drop%20at%20night

Macdonald, L.W. (2007). Fatal Flaws: Uncertainty in the Interpretation of Colour in CCTV Images. *Annals of the British Machine Vision Association (BMVA) 2007,* (7) 1-11. http://www.bmva.org/annals/2007/2007-0007.pdf

Wang, J., Hu, Y. (2019). An Improved Enhancement Algorithm Based on CNN Applicable for Weak Contrast Images. *IEEE Access, 8,* 8459-8476, 2020. https://doi.org/10.1109/ACCESS.2019.2963478

Hu L, Qin M, Zhang F, Du Z, Liu R. (2021). RSCNN: A CNN-Based Method to Enhance Low-Light Remote-Sensing Images. *Remote Sensing, 13*(1). https://doi.org/10.3390/rs13010062

Wang, T., Liu, M., Zhu, J., Tao, A., Kautz, J., Catanzaro, B. (2017). High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. https://doi.org/10.48550/arXiv.1711.11585

Yasenko, L., Klyatchenko Y., Tarasenko-Klyatchenko, O. (2020). Image Noise Reduction by Denoising Autoencoder. *IEEE 11th International Conference of Dependable Systems, Services and Technologies (DESSERT),* 351-355, 2020. https://doi.org/10.1109/DESSERT50317.2020.9125027

Mark, S. (2021). Day-night Dataset. *Kaggle.com*. https://www.kaggle.com/datasets/stevemark/daynight-dataset

Schwartz, E., Giryes, R., Bronstein, A. (2018). DeepISP: Learning End-to-End Image Processing Pipeline. https://www.kaggle.com/datasets/knn165897/s7-isp-dataset

Wei, C., Wang, W., Yang, W., Liu, J. (2018) Deep Retinex Decomposition for Low-Light Enhancement. *British Machine Vision Conference*, 2018. https://doi.org/10.48550/arXiv.1808.04560

Bank, D., Koenigstein, N., Giryes, R. (2020). Autoencoders. https://doi.org/10.48550/arXiv.2003.05991

Cunningham, J., Shu, D., Simpson, T.W., Tucker, C.S. (2020). A Sparsity Preserving Genetic Algorithm for Extracting Diverse Functional 3D Designs from Deep Generative Neural Networks. *Design Science, 6, E11*, 2020. https://doi.org/10.1017/dsj.2020.9

Hermosilla, G., Tapia, D. I. H., Allende-Cid, H., Castro, G.F., Vera, E. (2021). Thermal Face Generation Using StyleGAN. *IEEE Access, 9,* 80511-80523, 2021. https://doi.org/10.1109/ACCESS.2021.3085423.

Bakurov I., Buzzelli M., Schettini R., Castelli, Vanneschi, L. (2022). Structural Similarity Index (SSIM) Revisited: A Data-Driven Approach. *Expert Syst. Appl. 189*, 2022. https://doi.org/10.1016/j.eswa.2021.116087

Joshi K., Yadav R., Allwadhi S. (2016). PSNR and MSE Based Investigation of LSB. *International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT)*, 280-285, 2016. https://doi.org/10.1109/ICCTICT.2016.7514593

Zhang, L., Zhang, L., Mou, X., Zhang, D. (2011). FSIM: A Feature Similarity Index for Image Quality Assessment. *IEEE Transactions on Image Processing, 20*(8), 2378-2386, 2011. https://doi.org/10.1109/TIP.2011.2109730

Chen C., Chen, Q., Xu J., Koltun V. (2018). Learning to See in the Dark. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. https://doi.org/10.48550/arXiv.1805.01934

Shinde K., Hirve R., Kale M. (2021). Trying to See Low Exposure Images using CNN. *International Journal of Engineering Research & Technology (IJERT), 10*(5). https://www.ijert.org/research/trying-to-see-low-exposure-images-using-cnn-IJERTV10IS050126.pdf