

**Department of Electrical Engineering**

Project Name:

Image Classification in Low Light

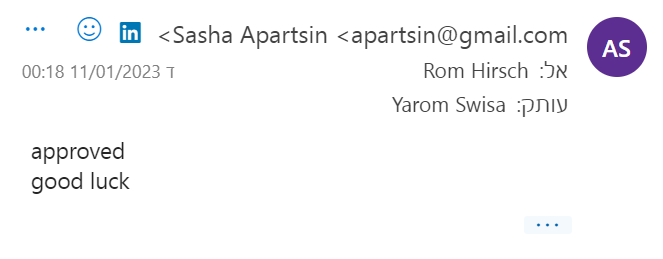
Student names: Rom Hirsch and Yarom Swisa

ID: 313288763 and 203675814

Mentor: Dr. Sasha Apartsin

Date of Submission: 20/01/2023

Mentor approval:



**Table of Contents**

[1. Introduction](#_1fob9te) 6

[2. Project Objectives](#_2et92p0) 8

[3. Success Metrics](#_tyjcwt) 8

[3.1. Photo-realistic algorithm](#_3dy6vkm) 8

[3.2. Classification measures](#_1t3h5sf) 8

[3.2.1. Classification Accuracy](#_4d34og8) 8

[3.2.2. Precision](#_2s8eyo1) 9

[3.2.3. Recall](#_17dp8vu) 9

[3.2.4. F1-score](#_3rdcrjn) 9

[4. Literature review](#_gi4z46jmrupl) 10

[4.1. Low-Light Image Enhancement Methods](#_lnxbz9) 10

[4.2. Low-Light image augmentation](#_92a7yl3mlt65) 10

[4.3. Deep learning classification methods](#_j5yvkoounkpw) 11

[5. Methods](#_leepqfg0hn99) 12

[5.1. Block Diagram (Experiment and Implementation workflow)](#_44sinio) 12

[5.2. Description of Project Phases](#_z337ya) 12

[5.3. Project phases in depth](#_s9le2nglhut2) 14

[5.3.1. Phase 1 – Data augmentation](#_4exiktx8iluz) 14

[5.3.1.1. Datasets](#_bhmtt1fx07md) 14

[5.3.1.2. Augment photo realistic low-light images](#_cfn082mjaepn) 15

[5.3.1.3. Evaluation and turning parameters](#_a4rn6wwvzuj0) 18

[5.3.1.4. Evaluation](#_niey4jvup1qj) 19

[5.3.1.5. Low light augmentation levels](#_t3lwr8bi6hd9) 20

[5.3.2. Phase 2 – Low-light model benchmark](#_izbehkosjw4i) 21

[5.3.2.1. Classification model](#_kxy3vh2pqvqg) 21

[5.3.2.2. Training approach](#_jjqe37l3olg4) 21

[5.3.2.3. Hyper parameters tuning](#_jptykr1rdzeg) 21

[5.3.2.4. Preprocessing - standard augmentation](#_8j1u3v1l1590) 22

[5.3.2.5. Evaluate accuracy for pristine images](#_8zpz9sezf65n) 22

[5.3.2.6. Evaluate loss of accuracy for low light images](#_ays3wqxal62h) 22

[5.3.3. Phase 3 - image enhancement](#_h9qmnhx9b6tc) 22

[5.3.3.1. Histogram equalization](#_6o639q6izsad) 22

[5.3.3.2. Adaptive histogram equalization:](#_x0vcobt1dymb) 23

[5.3.3.3. Dynamic histogram equalization](#_i39adx70dwm2) 24

[5.3.3.5. Retinex image enhancement](#_5w5uybprjo6p) 25

[5.3.4. Phase 4 – Image augmentation approach](#_65dnx1mjtwvs) 26

[6. Preliminary results](#_o51yfwyu33yo) 27

[6.1. Low light augmentation](#_d6krwhh1dnhl) 27

[6.1.1. Evaluation of augmentation algorithm](#_ep6nlbxlptyy) 27

[6.1.2. Compare augmented images with real dark images](#_b41n1fnhfjkp) 30

[6.2. Trained model accuracy on pristine images](#_5xtjrv5dihbg) 32

[6.3. Accuracy loss for dark images](#_tzqvretfb5lu) 33

[6.4. Model training with pristine and dark images](#_2m4umfqilk5w) 34

[6.4.1. Model training with real dark images](#_67vw0gv05o5x) 34

[6.4.2. Model training with augmented dark images](#_5ubz7i619zid) 35

[6.5. Image enhancement](#_87qut0u1dp0f) 39

[6.5.1. Histogram equalization](#_l5peyucvzb3b) 39

[6.5.2. Adaptive histogram equalization](#_57qjl9g0gj65) 39

[6.6. Partial conclusions](#_n31398uct91e) 40

[7. Engineering challenges](#_69pzq4exir5) 40

[8. Division of work between the partners:](#_1y810tw) 40

[9. Required tools](#_2xcytpi) 41

[9.1. Programs language](#_1ci93xb) 41

[9.2. Develop environments and data analysis:](#_3whwml4) 42

[10. Expected outcomes/deliverables](#_2bn6wsx) 43

[11. Work plan](#_tk1j36orrkak) 44

[12. References](#_1pxezwc) 46

[12. Appendices](#_qb8bclrl01jf) 47

[12.1 Code](#_ip55lmr53vke) 47

[12.1.1 Interesting functions and classes](#_4gavru1d79d0) 47

[12.1.1.1 Augmentation algorithm](#_hakmf2padeh7) 47

[12.1.1.2 Deep learning class](#_irg434q0p5st) 50

[12.1.1.3 DPhandler.py main](#_e3revbflsmfa) 50

[12.1.1.4 Dark\_images\_analyze.py and ds\_analyze.py](#_c4fpgb8y6yy) 50

Abstract

Accurate image classification is a key requirement in many computer vision systems. One of the most critical factors for object classification accuracy is illumination conditions under which the images are acquired from sensors. In many practical applications including automotive, video surveillance and photography, object classification for low light images is required.

Object classification becomes more challenging in low light images due to various image degradation resulting from low light acquisition conditions: color bias, unknown noise, detail loss and halo artifacts. The conventional deep learning models frequently underperform on low light images since low light images have different properties compared to the images that were used during the model training phase.

The objective of the project is to evaluate different methods for improving the classification accuracy in low light images including image enhancement prior to classification and training Deep Learning models with low light images synthetically generated from pristine image dataset.

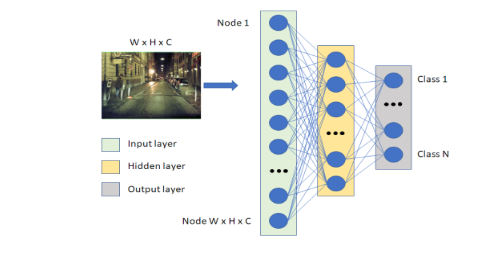
# 

# 1. Introduction

Image classification is a main element in many computer vision systems. Robust image classification algorithms are important in many areas, such as traffic monitoring, collision avoidance, face recognition, etc.

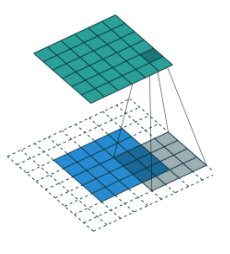
**Classification** is a process of assigning a class to the context of an image. Some of the main classification methods based on Machine learning algorithms include Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and logistic Regression. The biggest advantage of these classifiers is their ability to perform classification by using relatively small datasets. However, for larger datasets and complex problems they are limited. In order to deal with large datasets and more complex problems there are classification methods based on Neural Network (NN), NN provides good performance while working with large datasets.

**Neural Network (NN)** is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. NN rely on training data to learn and improve their accuracy over time. NN architecture includes an input layer, hidden layer, and an output layer (figure 1).

[](#D2L_fig_label_NN - example)  
Figure 1: NN - example

In case of **image classification,** the input dimension usually defined as Width\*Height x 3 for RGB and Width\*Height x 1 for gray scale format. Hence the number of calculations that NN should perform in order to provide classification depends strictly on the image size. Extensive calculations are the main factor that limits the possibilities of NN to perform classification and detection on large size images. The most common methods for image classification are based on Convolutional Neural Network (CNN) architecture.

**Convolutional Neural Network (CNN)** is using a convolution in place of general matrix multiplication in their layers (figure 2). They are specifically designed to process pixel data and are used in image classification. Image classification with CNN works by sliding a kernel across the input image to capture relevant details in the form of features.

[**](#D2L_fig_label_ convolution operation) *Figure 2: convolution operation*

One of the problems image classifications need to deal with is Low-light images which we encounter in many modern systems, such as surveillance, and autonomous driving etc. Unfortunately, low light images Classification is a challenging task, low-light conditions are not only low in brightness, but they also suffer from many other problems such as color bias, unknown noise, detail loss and halo artifacts.

There are some methods to overcome this problem. The first is to enhance the low light images before passing through the classifier model and the second is to train the classification NN model on a data set that includes low light images in the first place.

In this project we will use advanced low-light images enhancement methods and evaluate their contribution to the classification accuracy. The results will be compared with a model that was trained from the beginning for low-light images. In our project we will be focused on the goal of low-light images classification accuracy improvement. Our main research questions are as follows:

**RQ1:** What low-light image enhancement methods can be used to improve classification accuracy of deep learning models?

**RQ2:** Does including low-light images in the training set improve classification accuracy of deep learning models?

**RQ3:** Which is the best method for improving the classification accuracy of deep learning models?

# 

# 2. Project Objectives

The objective of this project is to implement and compare two different approaches of dealing with low light image classification:

* **Approach A**: Apply image enhancement before using the conventional image classification model.
* **Approach B**: Train the image classification model using synthetic low light images as part of the training dataset

We will compare the results of the different approaches and evaluate the best solution. For achieving the above adjective, the following goals must be met

* **G1**: Design, implement and validate a photo-realistic algorithm for generation of synthetic low light images.
* **G2**: Research, select and implement existing image enhancement algorithms for improving low light images
* **G3**: Design and train a new image classification model with the help of synthetic low light images
* **G4**: Design and execute the performance analysis and comparison study

# 3. Success Metrics

## 3.1. Photo-realistic algorithm

In order to evaluate the algorithm, we will collect real low light images. In addition to visual comparison We will analyze the illumination distribution of real low-light images and check the illumination distribution of the synthetic fitting to the real low light images.

## 3.2. Classification measures

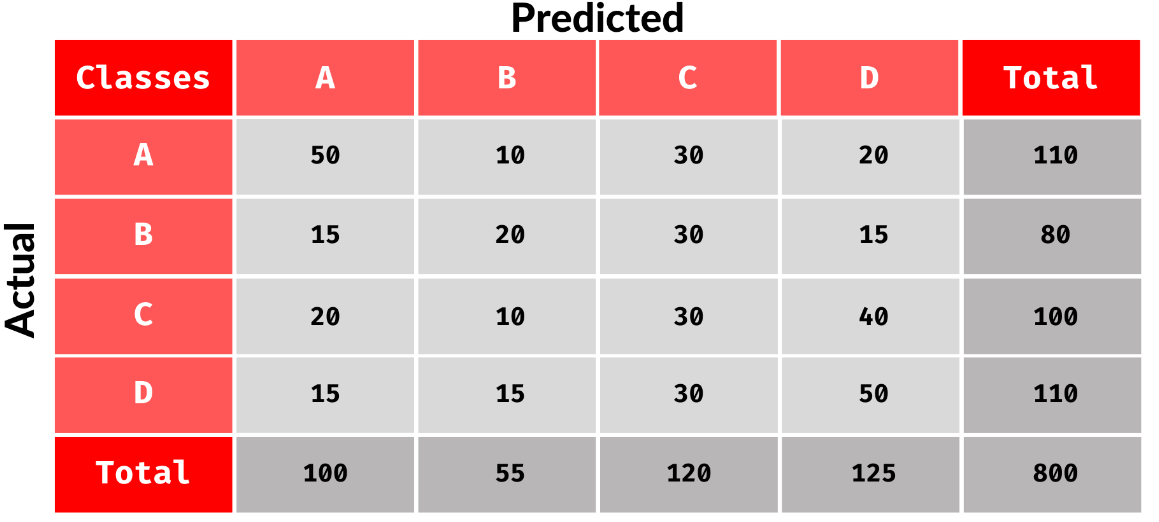
The different methods for classification will be measured by 4 measures:

* accuracy
* Precision
* Recall
* F1-score

These measures will guide us to conclude the best approach for low light image classification.

### 3.2.1. Classification Accuracy

Classification accuracy can be calculated from a confusion matrix. A confusion matrix is a tabular way of visualizing the performance of your prediction model. Each entry in a confusion matrix denotes the number of predictions made by the model where it classified the classes correctly or incorrectly.

[](#D2L_fig_label_Confusion matrix)  
Figure 3: Confusion matrix

The accuracy is defined as Accuracy = (TP + TN) / (TP + TN + FP + FN) and is calculated per class where:

* True Positive (TP): defined as how many images were classified correctly (the actual value and the predicted value are the same).
* False Negative (FN): defined as the amount of times the model miss predicted the class (the sum of values of corresponding rows except the TP value).
* False Positive (FP): defined as the amount of times the model images as the class and was wrong (The sum of values of corresponding column except TP value).
* True Negative (TN) : defined as the number of times the model predicted the images as other classes and was right (the rest of the values).

### 3.2.2. Precision

It tells you what fraction of predictions as a positive class were actually positive. Precision = TP/(TP+FP)

### 3.2.3. Recall

It tells you what fraction of all positive samples were correctly predicted as positive by the classifier. It is also known as True Positive Rate (TPR), Sensitivity, Probability of Detection. Recall = TP/(TP+FN)

### 3.2.4. F1-score

It combines precision and recall into a single measure. Mathematically it’s the harmonic mean of precision and recall. F1-Score = 2\*Precision\*Recall/(Precision + Recall). Because we have multi class problem we used macro f1 score define as:

Macro F1 score is the unweighted mean of the F1 scores calculated per class. It is the simplest aggregation for F1 score.

macro f1 score = sum(F1 score) / number of classes

### 3.2.5. Improvement ratio

We define improvement ratio as:

improvment ratio = actually improved/ potential improvement

When:

actually improved = measured preformence (Accuracy) - baseline preformence

potential improvement = baseline preformence (Accuracy) - degradation performance

baseline preformence (table X) - pristine dataset performance of DP trained on pristine images

degradation performance - tested dataset performance of DP trained on pristine images tasted.

## 3.3. Enhancement methods

### 3.3.1. SSIM

Structural Similarity Index Measure (SSIM) is a widely used method to measure the similarity between two images. The SSIM measure compares the structural information in the two images rather than just pixel values. It is designed to take into account the human visual system's perception of image quality.

The formula for calculating SSIM between two images:



# Where, C1 = (k1 \* L)^2 and C2 = (k2 \* L)^2, L is the dynamic range of pixel intensities (L = 255 for an 8-bit image), k1=0.01 and k2=0.03 by default.4. Literature review

## 4.1. Low-Light Image Enhancement Methods

Low light Image enhancement methods can be partitioned into three main categories. The first category methods are based on the histogram equalization (HE) (contrast adjustment using the images histogram). Dynamic histogram equalization (DHE) [1] - divides the histogram of the image into sub blocks and uses HE to stretch the contrast for each subblock. Adaptive histogram equalization (AHE) [2] - changes image contrast by calculating the histogram of multiple local areas of the image and redistributing the brightness.

The second category methods are Methods based on Retinex Theory [3] that assumes that an image is a combination of reflection and illumination. Such methods maintain the consistency of the reflectance, increase the brightness of the illumination, and take the pixel-wise product to enhance the low-light image. Low-light Image Enhancement via Illumination Map Estimation (LIME) [4] enhances a low-light image by estimating its illumination map. Single Scale Retinex (SSR) [5] aims to restore the brightness after Retinex decomposition. Multi-Scale Retinex (MSR) [6] combines the filtering results of multiple scales based on SSR, MSR adds a color recovery factor to tackle the color distortion caused by contrast enhancement in local areas of the image.

The third category is deep learning based methods– a great number of state-of-the-art methods have been developed for low-light image enhancement such as:LLNET [7] - a DP model for enhancing lightness and denoising images.

Branch Low-Light Enhancement Network (MBLLEN)][8] uses multiple subnets for enhancement and generates the output image through multi-branch fusion.

RetinexNet [9] - decomposes low-light input into reflectance and illumination and enhances the lightness over illumination.

## 4.2. Low-Light image augmentation

We reviewed possible solutions for the problem of generating photo realistic low light images that includes: illumination reduction based on local and global change [16], cycleGan model for day to night transformation[17] and traditional image processing (gamma correction, contrast, brightness, blur etc.). We tested the above options and found that the best method that simulates low light images, based on visual, illumination histogram, contrast and brightness comparison to real low light images was the traditional image processing approach, based on article [15].

## 

## 4.3. Deep learning classification methods

Deep Residual Learning for Image Recognition (ResNet) [10] is a common neural network (NN) architecture used for deep learning computer vision applications like object detection, image segmentation and image classification. This network uses a technique called skip connections. The skip connection skips training from a few layers and connects directly to the output (figure 3).

The approach behind this network is instead of layers learning the underlying mapping, we allow the network to fit the residual mapping. This architecture allows a depth of up to 150+ layers that allow high accuracy and led to a breakthrough in the field of classification and identification accuracy. The method is still relevant and there are various improved methods of it such as ResNet200 [11]. Rethinking Model Scaling for Convolutional Neural Networks (EfficientNet) is a convolutional neural network architecture and scaling method that uniformly scales all dimensions of depth/width/resolution using a compoundcoefficient. Unlike conventional practice that arbitrarily scales these factors, the EfficientNet scaling method uniformly scales network width, depth, and resolution with a set of fixed scaling coefficients. EfficientNets also transfer well and achieve state-of-the-art accuracy on CIFAR-100 (91.7%), Flowers (98.8%), and 3 other transfer learning datasets, with an order of magnitude fewer parameters. EfficientNetv2 [18] is an improvement over the original EfficientNet model, which was introduced by Google Research in 2019. EfficientNetv2 is based on the idea of scaling up CNNs in a more structured and efficient way, by carefully balancing the network width, depth, and resolution. This allows EfficientNetv2 to achieve better accuracy with fewer parameters and lower computational complexity compared to other models of similar capacity. One key feature of EfficientNetv2 is its use of compound scaling, which scales the network in multiple dimensions (width, depth, and resolution) simultaneously. This helps to improve the model's representation power and efficiency. Additionally, EfficientNetv2 uses a modified version of the MBConv block, which is a building block for CNNs that consists of depthwise convolutions and pointwise convolutions, to further improve the model's efficiency and performance. Overall, EfficientNetv2 is a powerful and efficient model that has achieved state-of-the-art performance on a variety of image classification tasks.

# 

# 

# 

# 5. Methods

## 5.1. Block Diagram (Experiment and Implementation workflow)

1. Data augmentation

2. Low-light model benchmark

Evaluate accuracy on pristine images

Estimate accuracy loss on augmented images

Select dataset

Augment photo realistic low-light images





3. Image enhancement approach

4. Image augmentation approach

Evaluate accuracy improvement

Implement enhancement methods

Evaluate accuracy improvement

Implement and train





5. Comparative analysis and recommendation

## 5.2. Description of Project Phases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Title** | **Objectives** | **Subtasks** | **Expected outcome** |
| **1** | **Data augmentation** | Prepare dataset that contains synthetically generated low light images with known degradation parameters (e.g., SNR) | Select source image dataset | Low-light images dataset |
| Implement low light image augmentation model |
| Evaluate the quality of the augmentation |
| Generate augment dataset |
| **2** | **Low-light model benchmark** | Evaluate how state-of-the-art conventional models deal with low light images. Find the dependency between degradation parameters and loss of the accuracy | Evaluate accuracy on pristine images   * develop setup environment for deep learning * Train the model on the pristine images * Test the accuracy of the model for the pristine images | Baseline accuracy on pristine and low-light images of model |
| Estimate accuracy loss as function of degradation parameters |
| **3** | **Image enhancement approach** | Evaluate the improvement over benchmark that might be obtained by image enhancement applied prior to image classification as function of enhancement parameters and image degradation parameters | Research/Implement image enhancement methods   * Review the different solutions for low light image enhancement * Implement the chosen methods | Accuracy improvement as function of enhancement images |
| Evaluate accuracy improvement as function of enhancement and degradation parameters |
| **4** | **Image augmentation approach** | Evaluate performance of the model trained with low light synthetic images included in the training dataset | Design and implement training procedure | Accuracy improvement over the benchmark as function of degradation and model parameters |
| Evaluate accuracy improvement over the benchmark as function of degradation and model parameters |
| **5** | **Comparative analysis and recommendation** | Compare two approaches and provide a recommendation over applicability of each approach depending on the expected image degradation levels | Compare two approaches for various combinations of parameters | Recommendations over applicability of each approach depending on the expected image degradation levels |
| Summarize and document recommendations |

[Table 1: Description of project phases](#D2L_table_label_Description of project phases)

## 5.3. Project phases in-depth

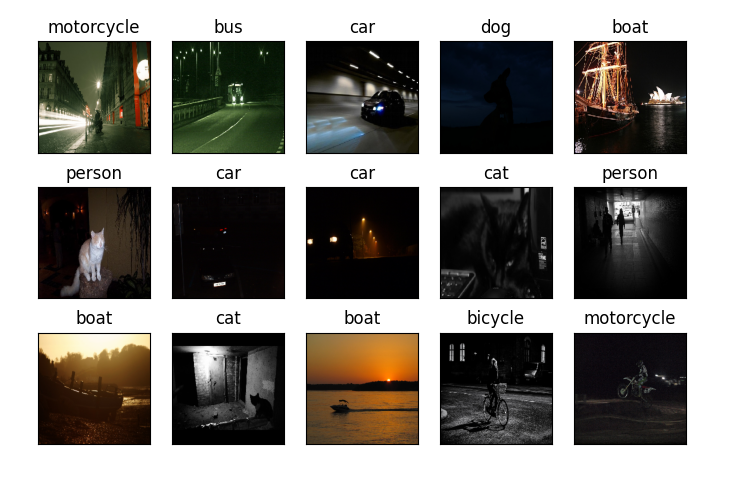
### 5.3.1. Phase 1 – Data augmentation

#### 5.3.1.1. Datasets

We used different datasets for a variety of tasks listed in table 2. In addition, we decided to extract sub-data from large datasets, this is because of our limited resource of hardware and time for training.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Name** | **Description** | **Sub dataset** | **Tasks** |
| **1** | LOL (LOw-Light) | The LOL dataset is composed of 500 low-light and normal-light image pairs and divided into 485 training pairs and 15 testing pairs. The low-light images contain noise produced during the photo capture process [21]. |  | Evaluate augmentation quality |
| **2** | Coco | The MS COCO (Microsoft Common Objects in Context) dataset is a large-scale object detection, segmentation, key-point detection, and captioning dataset[20]. fig.5 examples | Class: Bicycle, Boat, Bus, Car, Dog, Motorcycle, Person  Train\validation: 11664 images  Test: 1046 images | Fine-turning train and evaluate |
| **3** | ExDark | Exclusively Dark (ExDark) dataset (CVIU2019). The ExDark dataset is a collection of 7,363 low-light images. This dataset contains a variety of real low light images [19]. fig 4 examples | Class: Bicycle, Boat, Bus, Car, Dog, Motorcycle, Person  Train: 3860  Test: 1282 | Fine-turning train and evaluate the model on real low light images |

[Table 2:](#D2L_table_label_Datasets ) [Description of](#D2L_table_label_Description of project phases) [Datasets](#D2L_table_label_Datasets )

[](#D2L_fig_label_Examples ExDark )  
Figure 4: Examples ExDark

[](#D2L_fig_label_example COCO dataset)  
Figure 5: Example COCO dataset

#### 5.3.1.2. Augment photo realistic low-light images

We use a low light image simulation method to synthesize realistic low light images from normal light images based on the method suggested in [15]. Low light images differ from normal images due to the following dominant features: low brightness/contrast, noise, and sharpness. We found a transformation to convert the normal image to the low light image by analyzing images with different low light conditions. The article [15] proposes a combination of linear parameters that change the contrast and gamma correction that can offer good results. The low light image simulation pipeline (without additional noise and blurring) can be formulated as:

Where is the synthetic low light image. is the normal light image. α is a linear transformation that has an effect on contrast. is the gamma correction parameter used to correct\change the brightness of an image. is the contrast parameter before gamma correction. is contrast after gamma correction to get the image darker and blur with more difficult conditions.

As for noise we add Gaussian noise that simulates noise that happened due to the level of lighting. is variance

For Blurring we use a gaussian blur filter that smooth the images

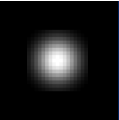


Figure 6: Gaussian window

Smoothing depend on – small results in little smoothing and vice versa.

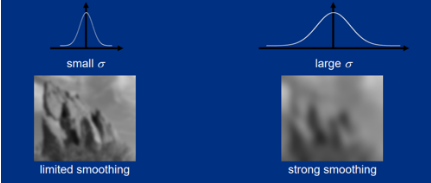
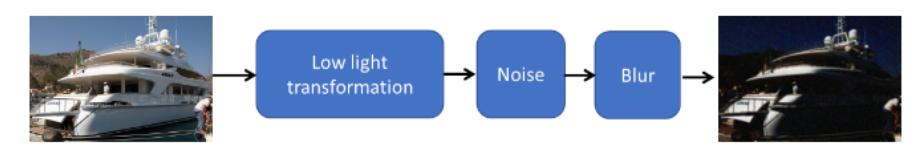


Figure 7: var effect

Figure 8: Augmented algorithm pipeline 

#### 

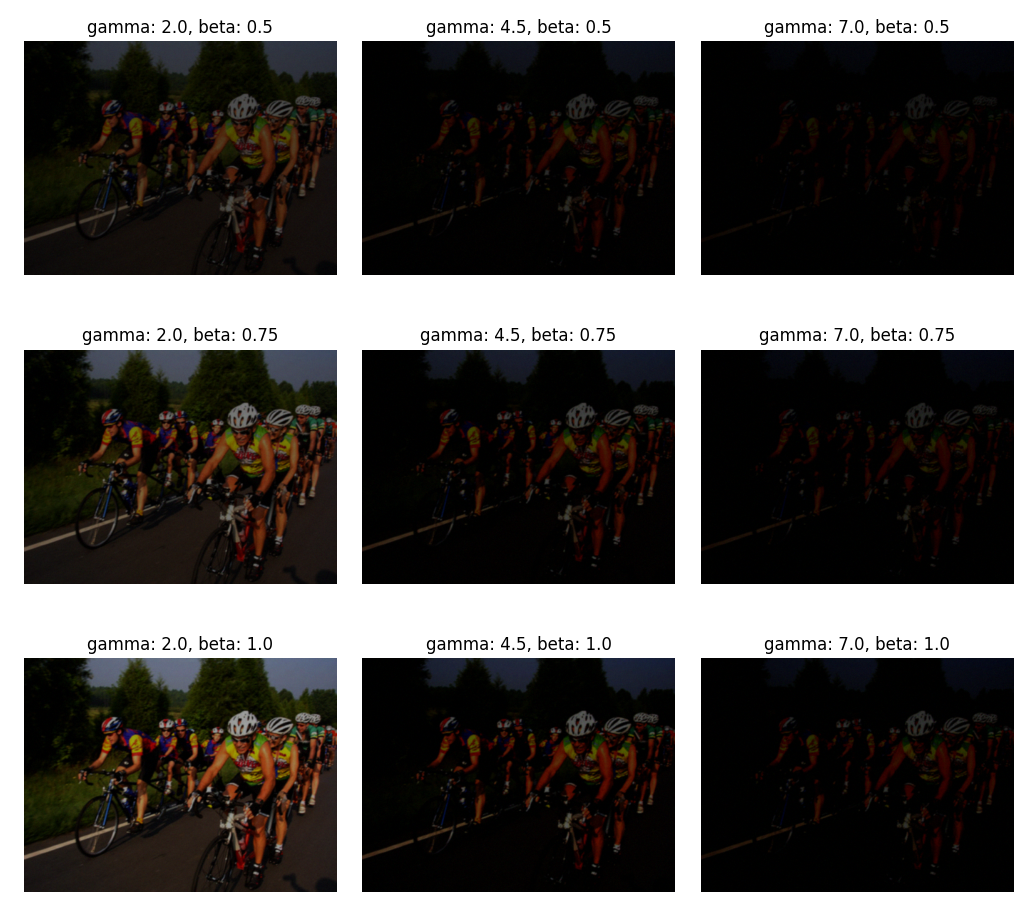
#### 

#### 5.3.1.3. Evaluation and turning parameters

By compare to real low light images we get the following distribution for the low light transformation parameters:

, 0.001, 0.01), ,

Figure 6 shows augmented images with different parameters with the mean brightness and contrast of our dataset. The columns are different gamma values and the rows are the different beta values where beta changes the contrast and gamma controls how dark the image is.

[](#D2L_fig_label_Augmented image example - gamma: columns, beta:rows)  
Figure 6: Augmented image example - gamma: columns, beta:rows

#### 5.3.1.4. Evaluation

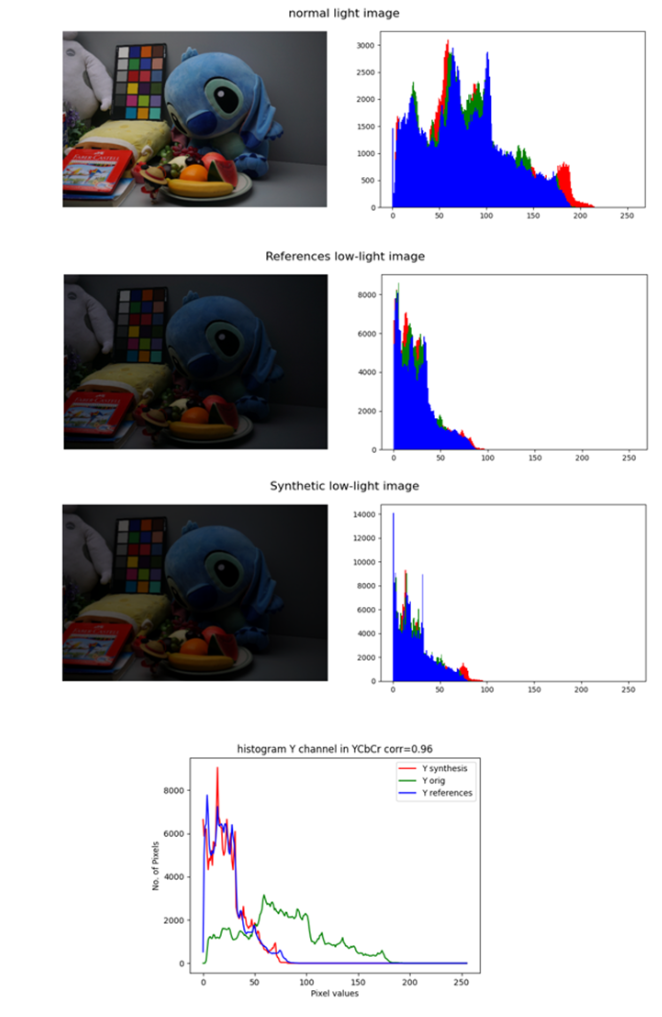
We verify the low-light transformation by comparing the algorithm output to real images taken in low-light conditions. We used visual comparison and histogram comparison of Y channel in YCbCr (Y is luminance\light intensity) between synthetic image and real dark image.

Figure 10: Evaluation examples - It can be seen that the histogram of the real dark image resembles the histogram of synthetically generated dark image

#### 5.3.1.5. Low light augmentation levels

We chose different parameters that will generate different levels of low light images by extracting the differences in those parameters from normal and low light images. For our testing we created 4 levels of dark images (high level more difficult conditions) with the distribution of the following parameters:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| Level 1 |  | 1e-5, 1e-2) |  |  |
| Level 2 |  | 1e-5, 1e-2) |  |  |
| Level 3 |  | 1e-5, 1e-2) |  |  |
| Level 4 |  | 1e-5, 1e-2) |  |  |

[Table 2: Augmentation levels parameters](#D2L_table_label_Augmentation levels parameters)



*Figure - normal images*

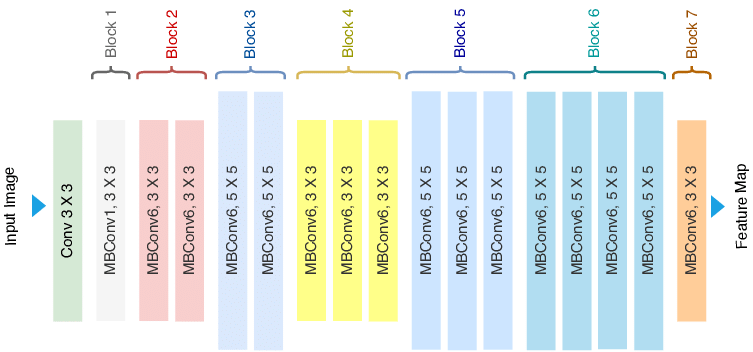


Figure 11: Example - 4 levels of dark images (high level more difficult conditions)

### 5.3.2. Phase 2 – Low-light model benchmark

#### 5.3.2.1. Classification model

We chose EfficientNetV2B0 pre-trained on ImageNet as our model because it is a convolutional neural network (CNN) architecture that has been shown to outperform many other popular CNN architectures on a wide range of benchmark datasets. EfficientNetV2B0 is commonly used in image classification.



#### 5.3.2.2. Training approach

We decided to approach the task by using fine-tuning because this technique is commonly used when the size of the new dataset is limited, As the pre-trained model provides a good starting point for the task and can help to improve the model performance. Fine-tuning typically involves adjusting the loss function, batch size, optimization algorithm, learning rate and fine-tuning some of the layers in the pre-trained model, rather than training the model from scratch. This can help to avoid overfitting on the new dataset and can often lead to better performance than training the model from scratch on the new data. The training setup of the experiments includes replacing the last classification layer of the pre-trained EfficientNetV2B0 model which has 1,000 object classes for the ImageNet into the 12 object classes of the experimented dataset

For our experiments, we train all the models with a Cross-entropy **loss function**, Cross-entropy is a common loss function for image classification. Cross-entropy loss measures the performance of a classification model whose output is a probability value between 0 and 1 for each class. Cross-entropy loss increases as the predicted probability diverge from the actual label. A perfect model would have a log loss of 0. As for **the optimization** we chose to use Adam (Adaptive Moment Estimation) is our optimization algorithm. It is an adaptive algorithm that uses an exponentially decaying average of past gradients to scale the learning rate. This allows it to adapt to the changing nature of the gradients as the model is trained and can help to improve convergence and avoid getting stuck in local minima. This allows Adam to be more computationally efficient and effective in training large and complex models.

#### 5.3.2.4. Preprocessing - standard augmentation

Data augmentation is a technique used to generate modified versions of the original data. This can help improve the performance of a machine learning model by providing it with additional training examples to learn from, which can help prevent overfitting and improve generalization to new data. We chose to use random horizontal and vertical flips and brightness range (only for pristine images) augmentation, this can help the model learn to recognize the same object in different orientations, which can be important for achieving good performance on real-world data where the orientation of objects may vary.

#### 5.3.2.3. hyperparameter optimization

We utilized the grid search technique to optimize hyperparameters and improve performance. Grid search is a popular hyperparameter optimization technique used in machine learning to systematically search for the best combination of hyperparameters for a given model. Hyperparameters are the parameters that cannot be learned by the model during training and must be set by the user before training the model. Grid search works by defining a grid of hyperparameter values to search over. The user specifies the range of values for each hyperparameter, and grid search creates a cartesian product of all possible combinations of values. The model is then trained and evaluated on each combination of hyperparameters using some performance metric. The combination of hyperparameters that produces the best performance on the validation set is then selected as the optimal set of hyperparameters for the model.

As part of our experiment, we used the grid search method to explore different combinations of hyperparameters, including the learning rate schedule, learning rate, and batch size. We evaluated a total of 110 combinations and selected the combination that resulted in the lowest validation loss for each dark ratio. We use this selected combination of hyperparameters to train our models.

|  |  |
| --- | --- |
| **Parameters** | **Value** |
| Learning rate scheduling | True, False |
| Learning rate | 1e-2, 1e-3, 1e-4 |
| batch size | 16, 32, 64 |
| Dark ratios | 0%, 20%, 40%, 60%, 80%, 100% |

[Table 3:](#D2L_table_label_Grid Search parameters - we explore the combination  of four hyperparameters: learning rate schedule, learning rate, batch size, and augmentation  ) Grid of hyperparameter values

|  |  |  |  |
| --- | --- | --- | --- |
| **dark ratio** | **learning rate** | **batch size** | **learning rate schedule** |
| 0% | 1.00E-04 | 16 | TRUE |
| 20% | 1.00E-04 | 16 | TRUE |
| 40% | 1.00E-04 | 32 | TRUE |
| 60% | 1.00E-05 | 16 | TRUE |
| 80% | 1.00E-04 | 32 | TRUE |
| 100% | 1.00E-04 | 32 | TRUE |

[Table 4:](#D2L_table_label_hyperparameters of our model) best-performing [hyperparameters per dark ratio](#D2L_table_label_hyperparameters of our model)

#### 5.3.2.5. Evaluate accuracy for pristine images

In order to evaluate our model on pristine images we used a pre-trained EfficientNetV2B0 model and replaced the final layer with a new layer of softmax with 8 classes. We fine-tuned the model with the COCO dataset to adapt the pre-trained features and classification layer to the new data. The dataset was divided as follows: 60% train, 20% validation, and 20% test.

#### 5.3.2.6. Evaluate loss of accuracy for low light images

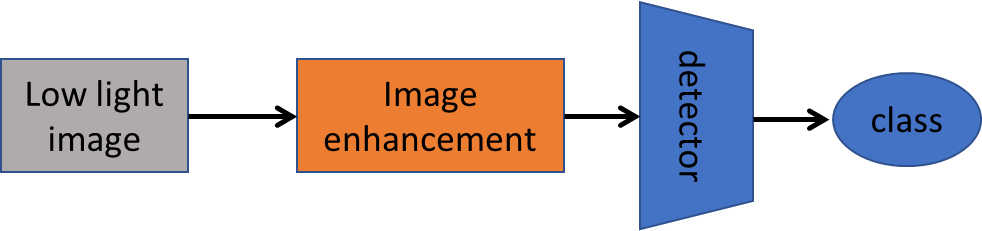
We tested our trained model (trained on pristine images) on augmented low light images with different levels as we defined and real low light images (ExDark dataset).

### 

### 

### 5.3.3. Phase 3 - image enhancement

In this phase, we are going to apply different image enhancement algorithms as another step in the preprocess and compare the results KPI’s to see if it improves the classification accuracy. We chose to use the following algorithms.

[](#D2L_fig_label_The proposed method consists of an image enhancement before base-line model (detector)  )  
Figure 7: The proposed method consists of an image enhancement before base-line model (detector)

#### 5.3.3.1. Histogram equalization

Histogram equalization is a method in image processing that is used to adjust the contrast of an image. It is used to stretch the intensity values of an image to cover the full range of possible values, which is typically from 0 to 255 for an 8-bit image. The goal of histogram equalization is to improve the contrast of the image, making it more visually appealing and easier to analyze. The process of histogram equalization involves calculating the histogram of an image. The histogram is then used to adjust the intensity values of the pixels in the image so that they are spread more evenly across the full range of possible values. This results in a more balanced and uniform distribution of pixel intensities, which can improve the contrast and visual appeal of the image. we are going to use cv2 equalization method, “[cv.equalizeHist](https://docs.opencv.org/4.x/d6/dc7/group__imgproc__hist.html#ga7e54091f0c937d49bf84152a16f76d6e)”

[](#D2L_fig_label_exmpale HE ExDark)  
Figure 8: exmpale HE ExDark

[](#D2L_fig_label_HE on augmented dataset)  
Figure 9: HE on augmented dataset

#### 5.3.3.2. Adaptive histogram equalization :

This is a method [1] of contrast enhancement in image processing that improves the contrast in images by adjusting the intensity values of the pixels in an image based on the image's local contrast. This is in contrast to traditional histogram equalization, which adjusts the intensity values of the pixels in an image based on the overall contrast of the entire image. AHE works by dividing the image into small blocks, called tiles, and performing histogram equalization on each tile separately. This results in improved contrast within each tile, but can also produce "blocking" artifacts at the boundaries between tiles. To mitigate these artifacts, the process can be repeated multiple times with progressively larger tiles, or the intensity values at the boundaries can be smoothed using interpolation or other techniques. We are using cv2 implementation (cv2.createCLAH).

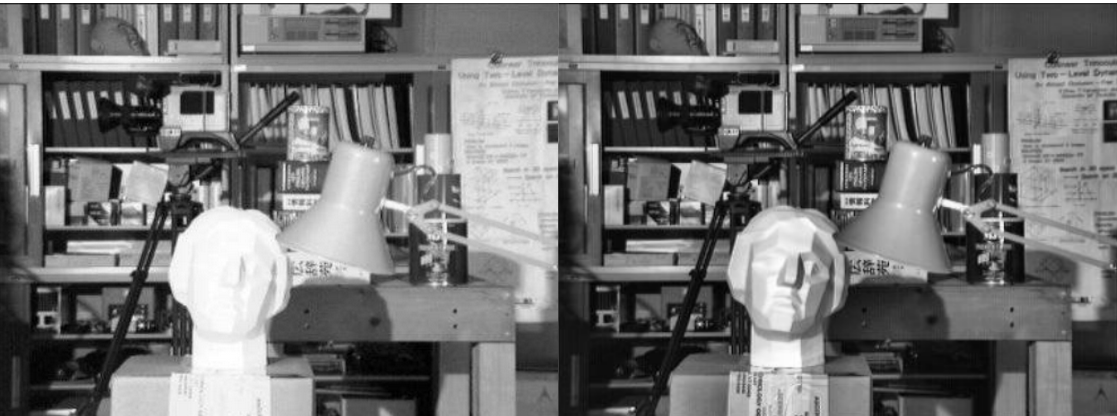
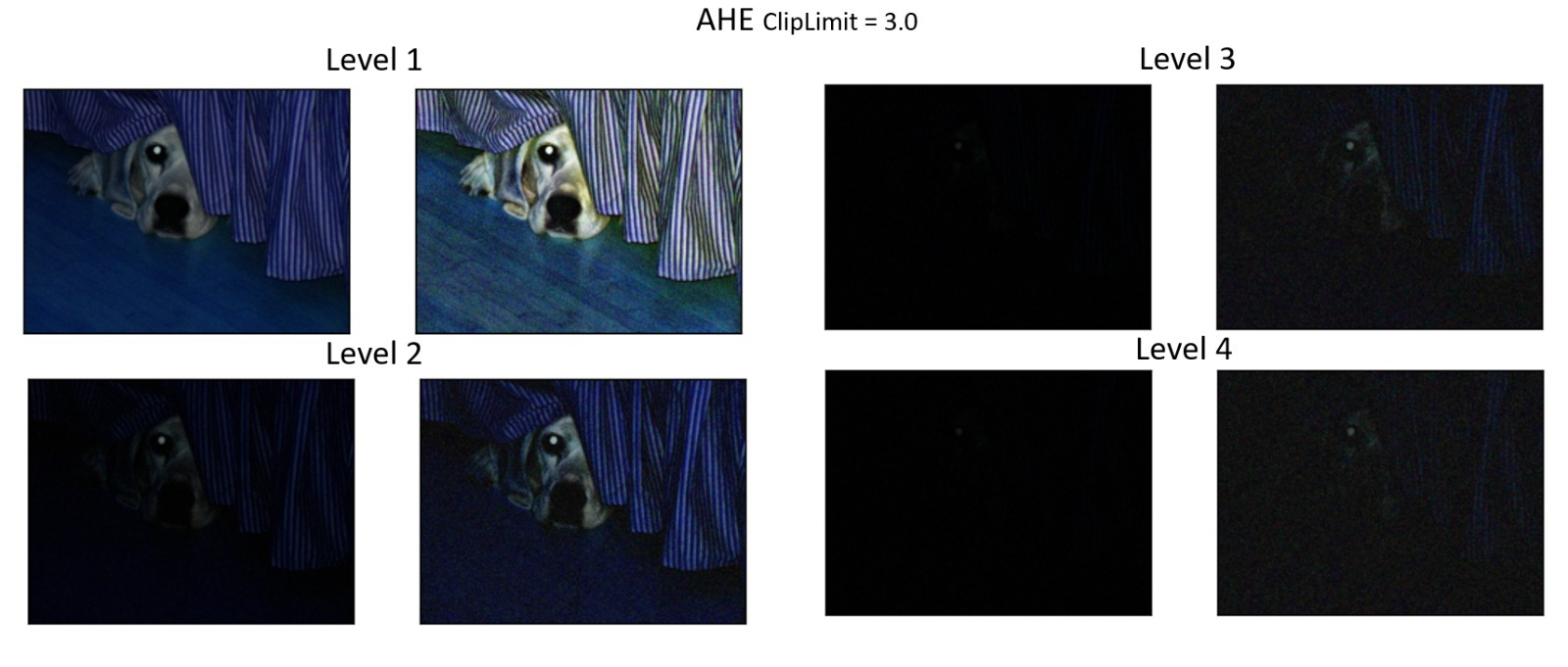
Figure 14: Example of AHE

Figure 15: AHE augmented datasets

#### 5.3.3.3. Dynamic histogram equalization

Dynamic histogram equalization (DHE) [2] is a method of contrast enhancement in image processing that adjusts the intensity values of the pixels in an image based on the local contrast of the image over time. This is in contrast to traditional histogram equalization, which adjusts the intensity values of the pixels in an image based on the overall contrast of the entire image, and adaptive histogram equalization (AHE), which adjusts the intensity values based on the local contrast within small blocks or tiles of the image. DHE works by dividing the image into small blocks, called tiles, and performing histogram equalization on each tile separately. However, rather than using a fixed set of intensity values for each tile, the intensity values are updated dynamically over time based on the current local contrast of the tile. This allows DHE to adapt to changes in the contrast of the image over time, such as those caused by changes in lighting or the movement of objects within the scene.

Figure 16: Example of DHE

#### 5.3.3.5. Retinex image enhancement methods

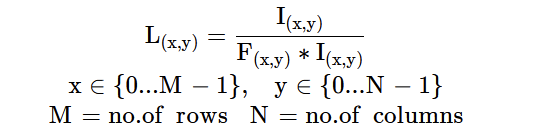
Retinex (Retina + Cortex) is a computational theory of vision that proposes that the human visual system uses two mechanisms to perceive the color and brightness of objects in an image: a local mechanism that extracts information from small regions of the image, and a global mechanism that extracts information from the entire image. Retinex image enhancement is a family of algorithms that use this theory to improve the color and brightness of images. These algorithms work by decomposing the image into its reflectance and illumination components, and then adjusting the illumination component to enhance the image's details and color while preserving its overall appearance.

Retinex(I)=Reflectance(r)∗Illumination(S)

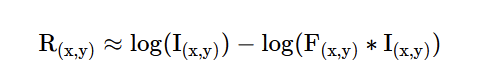
where I is the retinex image, r is the reflectance of the surface, and S is the illumination on the surface. And the lightness of an image, which is computed in our visual system (Retina + Cortex) is computed by reflectance and illumination. If we assume illumination is uniform or smooth, then lightness depends only on reflectance value at a given position. Computing the reflectance is a very high task and thus another method is suggested.

#### 5.3.3.6 Single-scale Retinex

The SSR [3] algorithm works by convolving the image with a Gaussian filter of a specific size. This filter acts as a smoothing operator, which removes high-frequency details from the image, leaving only the low-frequency components such as illumination and reflectance. The reflectance component is then enhanced by applying a logarithmic transform, which amplifies the low-intensity values and compresses the high-intensity values. The illumination component is also adjusted by applying a similar transform.



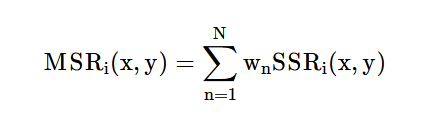
where L(x,y)​ is lightness of image at pixel position (x,y), and F(x,y)​∗I(x,y)​ is the average of the surrounding pixels at (x,y) given by the center surround function F(x,y)​.

As the retinex image is equal to the reflectance of the scene under no effect of illumination, the retinex image (R) is approximately equal to the relative lightness. That gives

If we consider Gaussian function (GσGσ​) as center-surround function, then retinex image for each i-th channel in an image is, SSRi(x,y)=log⁡(Ii(x,y))−log⁡(Gσ∗Ii)(x,y))

That is the Single-scale retinex of an image is estimated by taking the logarithm difference of image and point-surround filter of the image at position (x,y). The operation (Gσ∗Ii)(x,y) is nothing but gaussian blur of an image with given scale (σ).

#### 5.3.3.6 Multi-scale Retinex

Multiscale Retinex (MSR) [6] is an image enhancement algorithm that builds upon the Single Scale Retinex (SSR) algorithm. The MSR algorithm aims to improve the performance of SSR by allowing it to operate at multiple scales. In the MSR algorithm, the input image is first decomposed into multiple scales. Each scale contains a different level of detail, ranging from fine to coarse using a different filter. The SSR algorithm is then applied to each scale independently to obtain the reflectance and illumination components. However, unlike the SSR algorithm, MSR uses different scales of Gaussian filters to preserve the information from different levels of detail. After the reflectance and illumination components are obtained for each scale, they are combined to generate the final enhanced image. 

The MSR algorithm is particularly useful for enhancing images that contain details at different scales, such as images of natural scenes. Compared to SSR, the MSR algorithm provides more robust and efficient results. It can remove noise and improve contrast in the image while preserving its natural appearance.

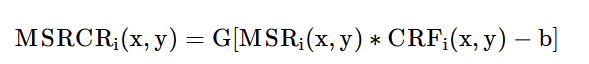
#### 5.3.3.7 Multi scale retinex with color restoration (MSRCR)

As the MSR [6] of an image looks colorless, the Multi-scale retinex output is multiplied with Color-restoration function (CRF) to restore the original colors of the input image approximately. And this method of calculating the Color-restoration function and applying it to Multi-scale retinex output is called Multi-scale retinex with Color Restoration (MSRCR).

where CRFi(x,y) is the color restoration value for pixel (x,y) at i-th channel. The color restoration function is defined as



where k equals no. of image channels, α is to control non-linearity, and β is to control total gain. To achieve better contrast results, the MSRCR equation is modified to include gain (G) and offset (b) values.



#### 5.3.3.7 Multi scale retinex with color preservation (MSRCP)

[6]....

#### 5.3.3.7 Lime

Low-light Image Enhancement via Illumination Map Estimation is an image enhancement technique used to improve the quality of images captured under low-light conditions. The method works by estimating an illumination map, which is used to enhance the image. The Illumination Map Estimation technique works by estimating the illumination map of the image, which represents the amount of light that illuminates each pixel. The illumination map is estimated using various Retinex-based algorithms. Once the illumination map is estimated, it can be used to enhance the image by increasing the brightness and contrast while reducing noise. This is achieved by dividing the image into two components: the illumination component and the reflectance component. The illumination component is then adjusted by multiplying it with a scaling factor, which is determined by the estimated illumination map. The reflectance component is then multiplied with the same scaling factor, which helps to preserve the natural appearance of the image.

#### 5.3.3.7 Multi-branch low-light enhancement network (MBLLEN)

MBLLEN is a novel method for low-light image enhancement by using deep learning technology. At the core of the method is a fully convolutional neural network. The MBLLEN consists of three types of modules, i.e., the feature extraction module (FEM), the enhancement module (EM) and the fusion module (FM). The idea is to learn to: 1) extract rich features up to different levels via FEM, 2) enhance the multi-level features respectively via EM and 3) obtain the final output by multi-branch fusion via FM. In this manner, the MBLLEN is able to improve the image quality from different aspects and accomplish the low-light enhancement task to its full extent.

#### 5.3.3.7 Semantically Contrastive Learning for Low-light Image Enhancement (SCL-LLE)

#### SCL-LLE is a method for image enhancement of low light images that uses 3 modules: low-light image enhancement network, a contrastive learning module and a semantic segmentation module. Beyond the existing LLE wisdom, it casts the image enhancement task as multi-task joint learning, where LLE is converted into three constraints of contrastive learning, semantic brightness constancy, and feature preservation for simultaneously ensuring the exposure, texture, and color consistency. SCL-LLE allows the LLE model to learn from unpaired positives (normal-light)/negatives (over/underexposed), and enables it to interact with the scene semantics to regularize the image enhancement network, yet the interaction of high-level semantic knowledge and the low-level signal prior is seldom investigated in previous methods.

#### 5.3.3.7 ZeroDCE++

DCE-NET stands for Deep Contrast Enhancement Network, which is a deep learning-based approach for enhancing low-light images. DCE-NET uses a convolutional neural network (CNN) to learn a mapping function between low-light and high-light images. It is trained using a dataset of pairs of low-light and high-light images, and the network is optimized to minimize the difference between the enhanced low-light image and the corresponding high-light image. DCE-NET employs a novel contrast loss function that enhances the contrast of the low-light image while preserving its details. The approach has shown promising results in enhancing low-light images, making them visually appealing and improving their quality for various computer vision applications.

#### 5.3.3.7 Visual comparison between the different methods

#### 

Figure 11: Visual comparison - compares the different image enhancement methods on different levels of dark image augmentations from least to most dark as input. From the comparison we can see how the different methods handle low light images. As we can see by visual comparison, deep learning based methods give more realistic results.

#### DL methods image enhancement

### 5.3.4. Phase 4 – Image augmentation approach

In this phase we mix dark images (augmented images) in the training set in different propositions to see if we can improve the models performance. In the image augmentation experiments, we use different ratios of bright to low light images, from 10:0 (only bright images) to 0:10 (only low-light images) maintaining the same overall number of training images, to fine-tune the model. We are also going to repeat this experiment with real dark images instead of augmented images and compare the results.

# 

# 6. Results

In the results, we will showcase the results of our proposed methods. In section 6.1, we will demonstrate that our synthetically generated dark images closely resemble real dark images based on selected metrics. Following that, we will demonstrate the accuracy achieved by our baseline model. In section 6.2, we will present the results obtained by our baseline model. Moving on, section 6.3 will illustrate the degradation of the baseline model on dark images datasets. In section 6.4, we will fine-tune our model by incorporating real dark images into the training process. This will allow us to evaluate the effectiveness of the augmentation method in an ideal scenario and estimate the potential improvement for the Exdark dataset. Finally, we will train our model with synthetic images and present the results in the chosen metrics. In section 6.5 we will introduce the improvement achieved by our baseline model with image enhancement methods.

## 6.1. Low light augmentation

### 6.1.1. Evaluation of augmentation algorithm by histogram

To evaluate our augmentation algorithm we compared the histogram of the illumination (Y(luminance\light intensity) channel of YCbCr) of real low light images and augmented low light images using Pearson cross correlation between the histograms and visual comparison. the average of the similarities (Pearson correlations) was an 85% match. The following figures show the RGB histograms of a normal image, a real dark image, and an augmented image of the original one. We can conclude the augmentation yielded images that are approximately close to real low-light images.

### 

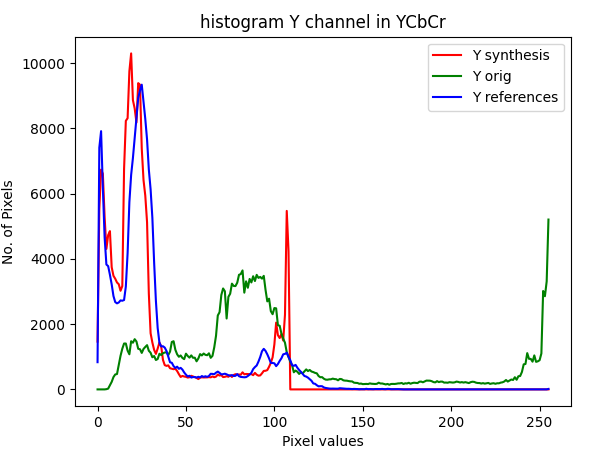


Figure 18: Example of histograms of the normal and augmented image - It can be seen that the histogram of the real dark image resembles the histogram of synthetically generated dark image.

### 

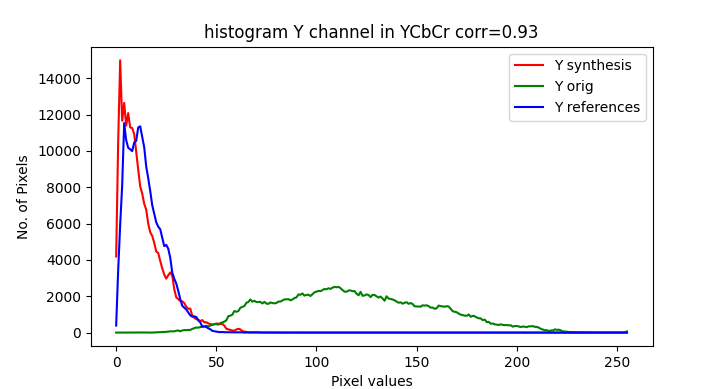
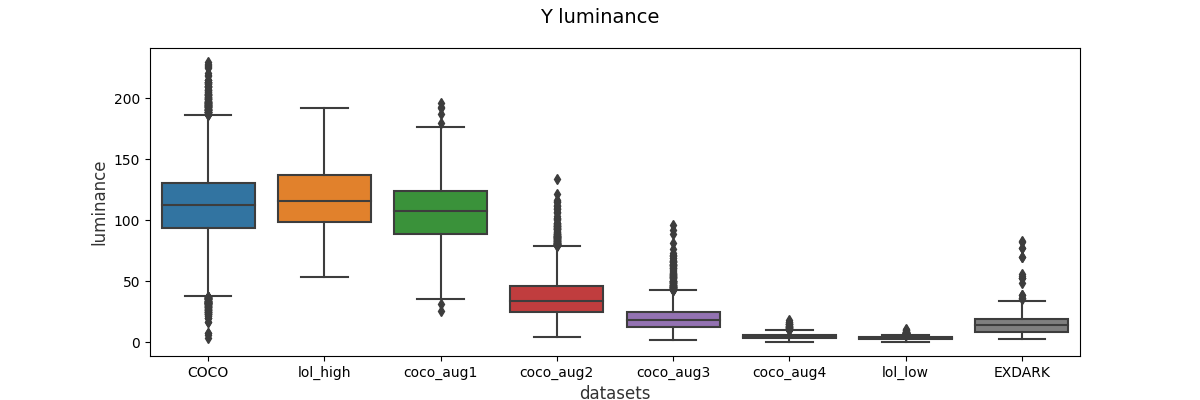
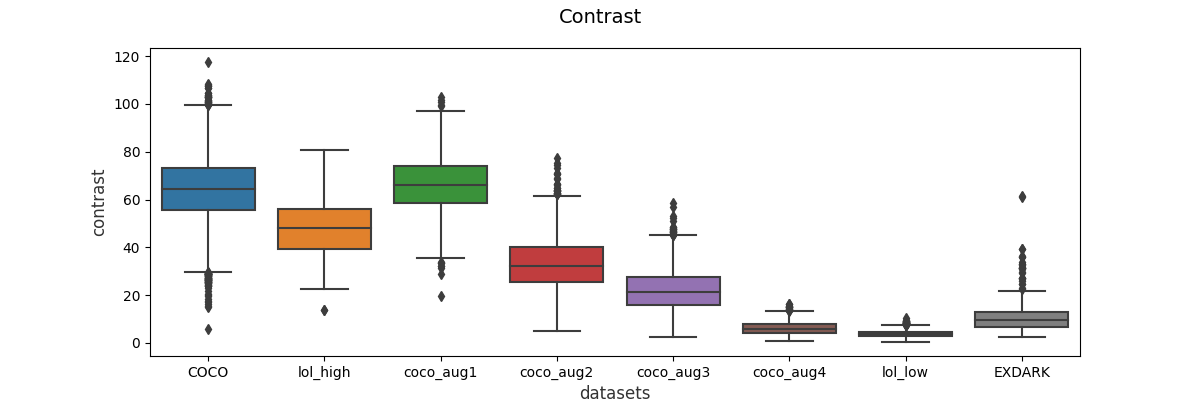
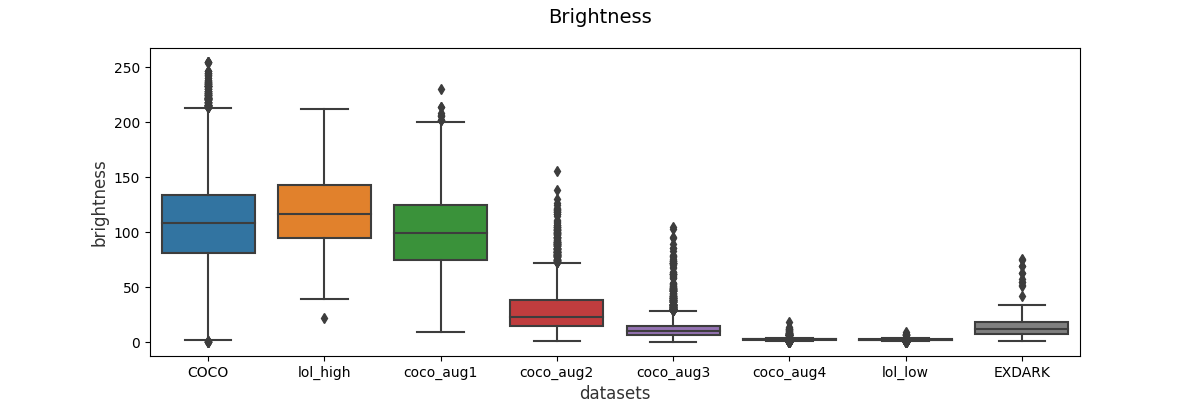


Figure 19: Example 2 of histograms of the normal and augmented image - It can be seen that the histogram of the real dark image resembles the histogram of synthetically generated dark image.

### 

### 6.1.2. Compare augmented images with real dark images

We evaluate the synthetically generated images, created from the LOL dataset which contains images with the same scenery both in normal and low light conditions, by comparing their contrast, average brightness, and saturation to the natural dark images. The following figures show boxplots of features we used to compare the datasets. Each figure shows the level of the features for COCO, COCO augmented in different levels, LOL normal and dark, ExDark. From the figures, we show that for the different features the augmented images are approximately similar to those of real dark images (augmented vs LOL low and ExDark). In the following graphs, we can compare COCO and LOL high as they are both images with normal light conditions. ExDark and LOL low are both data sets that contain real dark images and those will be our base line of comparison with our augmented images. As we can see we managed to create augmented data sets that show similar features distribution compared to the real dark images. COCO augmented 1 is more similar to the normal images and COCO augmented 2-4 have distributions that are similar to the real dark images data sets.

****

## 

Figure 20: Comparison of different features between all the datasets - augmented data sets show similar features distribution compared to the real dark images. COCO augmented 1 is more similar to the normal images and COCO augmented 2-4 have distributions that are similar to the real dark images (lol\_low and Exdark) data sets

## 

## 6.2 Baseline model training

## 6.2.2 Hyperparameter optimization

The hyperparameter optimization experiment involved utilizing the grid search approach to explore different sets of hyperparameters, which included the learning rate schedule, learning rate, and batch size. A total of 110 combinations were evaluated.

## The experiment includes the following steps:

1. Define the hyperparameters and their ranges ([Table 3:](#D2L_table_label_Grid Search parameters - we explore the combination  of four hyperparameters: learning rate schedule, learning rate, batch size, and augmentation  ) Grid of hyperparameter values) that will be used in the grid search
2. Perform the grid search and record the performance metric for each hyperparameter combination.

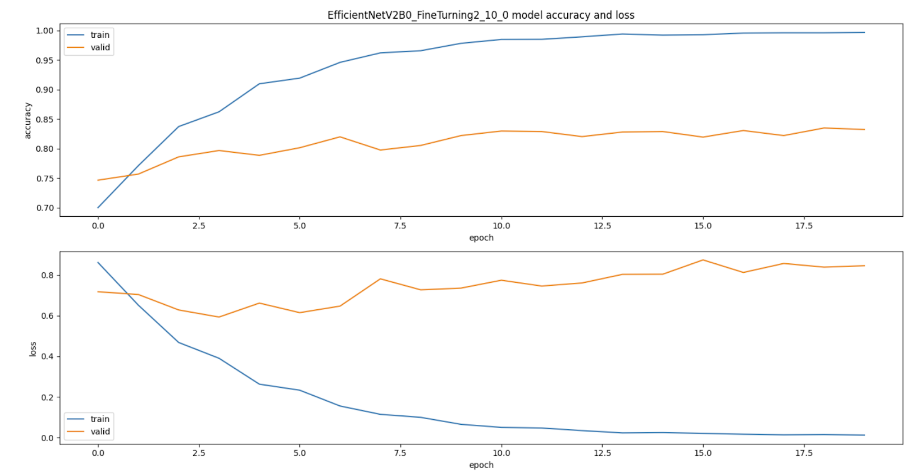
## We used the Parallel coordinates view (PCV) graph to select the best combination for each dark ratio model. The best hyperparameter describes in table 4 best-performing hyperparameter [per dark ratio](#D2L_table_label_hyperparameters of our model). PCV graph consists of a set of parallel axes, one for each hyperparameter, and a line connecting each hyperparameter combination based on their performance metric value. We selected for each dark ratio value the best hyperparamter combination (table 4).

## 

## 

## 6.2.2 Trained model accuracy on pristine images

We trained our model on the COCO dataset with fine-tuning. In order to track our models performance we used a loss curve and an accuracy curve. The loss curve shows the average loss, or error, of the model on the training and validation sets as the model is trained. The loss is a measure of how well the model is able to predict the correct output given an input. The lower the loss, the better the model is at making predictions. The accuracy curve shows the percentage of correct predictions made by the model on the training and validation sets as the model is trained. The higher the accuracy, the better the model is at making correct predictions. we can see from the figure below that as the training progresses the accuracy increases and the loss is decreasing and at the end they plateau.

Figure 21: model accuracy and loss

The table below shows the different metrics of our model trained and tested on coco dataset. the classification model shows high performance

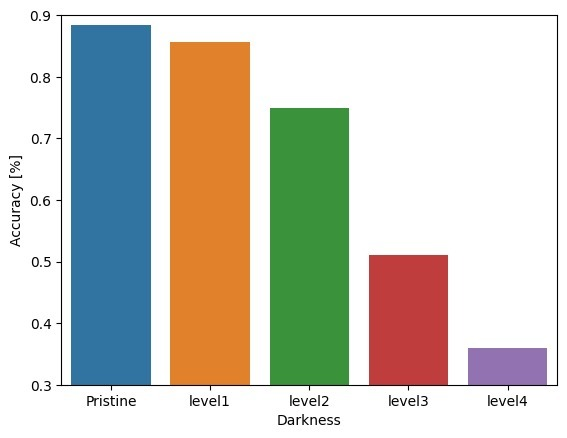
|  |  |  |  |
| --- | --- | --- | --- |
| **Accuracy** | **Precision** | **Recall** | **F1-score** |
| 0.884 | 0.895 | 0.882 | 0.882 |

[Table 3:](#D2L_table_label_Model metrics for normal images data set) Baseline performance: m[odel metrics for normal images data set](#D2L_table_label_Model metrics for normal images data set)

## 

## 6.3. Accuracy loss for dark images

We tested the model accuracy for low-light images, real and augmented. In the results, we can see that as the images get darker all the metrics show performance degradation.



(figure X): A plot showing how performance degraded when images were darker

| **Dataset** | **Accuracy** | **Precision** | **Recall** | **F1-score** |
| --- | --- | --- | --- | --- |
| Pristine (baseline) | 0.884 | 0.895 | 0.882 | 0.882 |
| ExDark | 0.661 | 0.727 | 0.658 | 0.668 |
| Augmented Level 1 | 0.857 | 0.877 | 0.854 | 0.857 |
| Augmented Level 2 | 0.75 | 0.787 | 0.748 | 0.752 |
| Augmented Level 3 | 0.51 | 0.601 | 0.505 | 0.514 |
| Augmented Level 4 | 0.36 | 0.484 | 0.354 | 0.353 |

[Table 4:](#D2L_table_label_Accuracy loss of the model for diffrent test sets) degradation preformence for [diffrent test sets](#D2L_table_label_Accuracy loss of the model for diffrent test sets)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ExDark** | **Level 1** | **Level 2** | **Level 3** | **Level 4** |
| 0.223 | 0.027 | 0.134 | 0.374 | 0.524 |

[Table 3:](#D2L_table_label_Model metrics for normal images data set) potential improvement

## 6.4. Model training with pristine and dark images

We designed experiments that train the model with different mixes of normal and dark images as such:

|  |  |  |
| --- | --- | --- |
| Model | Normal images % | Dark images % |
| #1 | 100 | 0 |
| #2 | 80 | 20 |
| #3 | 60 | 40 |
| #4 | 40 | 60 |
| #5 | 20 | 80 |
| #6 | 0 | 100 |

[Table 5: Models with diffrent training sets normal/dark images distribution](#D2L_table_label_Models with diffrent training sets normal/dark images distribution)

### 6.4.1. Model training with real dark images

In this experiment we mixed normal and real dark images as our training set. The following tables show that when combined with enough dark images the classification performances increase for dark images. From [Table 8](#D2L_table_ref_Metrics for COCO testset) we can see that we get performance degradation as we include more dark images in the training set. From table 4 we can see that including dark images in the training set greatly increases the model performance to classify dark images. The model with the best model performance is achieved by combining real dark images and therefore the model trained with augmented images could be as best as the 20/80 row in Table 4 (we used the results as baseline for improvement potential for ExDark dataset).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ratio (Normal/Dark) | Accuracy | Precision | Recall | F1-score |
| 100/0 | **0.884** | **0.895** | **0.882** | **0.882** |
| 80/20 | 0.794 | 0.817 | 0.793 | 0.792 |
| 60/40 | 0.83 | 0.839 | 0.829 | 0.83 |
| 40/60 | 0.803 | 0.828 | 0.801 | 0.801 |
| 20/80 | 0.787 | 0.799 | 0.785 | 0.789 |
| 0/100 | 0.658 | 0.688 | 0.656 | 0.644 |

[Table 8: Metrics for COCO](#D2L_table_label_Metrics for COCO testset) test set

Table 7 shows the results for all the models with different training set ratio metrics for normal images(COCO testset), we can see that the best performances are when the model is trained on only normal images.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ratio (Normal/Dark) | Accuracy | Precision | Recall | F1-score |
| 100/0 | 0.669 | 0.741 | 0.666 | 0.677 |
| 80/20 | 0.672 | 0.675 | 0.671 | 0.672 |
| 60/40 | 0.737 | 0.737 | 0.74 | 0.737 |
| 40/60 | 0.734 | 0.738 | 0.735 | 0.734 |
| 20/80 | **0.757** | **0.762** | **0.759** | **0.757** |
| 0/100 | 0.728 | 0.74 | 0.731 | 0.728 |

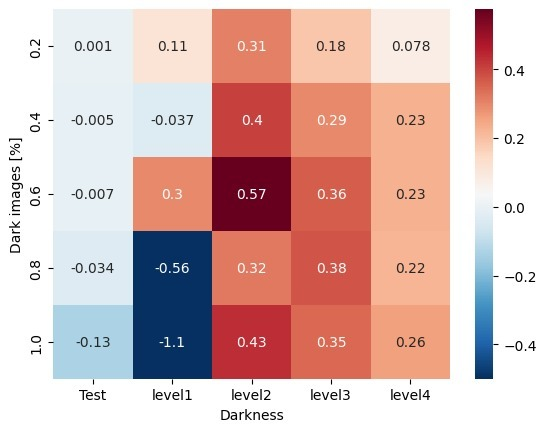
[Table 6: Metrics for ExDark testset](#D2L_table_label_Metrics for ExDark testset)

Table 8 shows the results for all the models with different training set ratio metrics for dark images(ExDark testset), we can see that the best performances are when the model is trained mostly on dark images (20% normal images and 80% dark images).

### 

### 6.4.2. Model training with augmented dark images

Our main experiments use different ratios of bright to low light images, from 10:0 (only normal images) to 0:10 (only low-light images) maintaining the same overall number of training images, to fine-tune different models and observe the classification outcomes on the same independent testing data and the test results are shown in the following tables 9-12 and heatmap X. from the heatmap we can see that there is clear dependency of performance between percentage of dark images in the training set and darkness level of test set. The overall best performance was yielded from using a training set of 60% dark images. From the heatmap left most column we can see that including dark images in the training set degrades the performance very slightly.

A figure showing the dependency of performance between (X-axis Darkness test set, Y axis - Darkness percentage in training set, hue axis - Improvement ratio) 

# 

The tables below are raw results of our experiments:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Accuracy | | | | | | |
| Model training set Ratio (Normal/Dark) | Test set | | | | | |
| COCO | ExDark | level 1 | level 2 | level 3 | level 4 |
| 100/0 | 0**.**884 | 0.669 | 0.857 | 0.75 | 0.51 | 0.36 |
| 80/20 | **0.885** | 0.695 | 0.86 | 0.791 | 0.579 | 0.401 |
| 60/40 | 0.879 | 0.693 | 0.856 | 0.804 | 0.62 | 0.479 |
| 40/60 | 0.877 | **0.71** | **0.865** | **0.827** | 0.645 | 0.48 |
| 20/80 | 0.85 | 0.654 | 0.842 | 0.793 | **0.651** | 0.476 |
| 0/100 | 0.75 | 0.637 | 0.827 | 0.808 | 0.64 | **0.496** |

[Table 8: Accuracy metric for test sets](#D2L_table_label_Accuracy metric for test sets)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| F1-score | | | | | | |
| Model training set Ratio (Normal/Dark) | Test set | | | | | |
| COCO | ExDark | level 1 | level 2 | level 3 | level 4 |
| 100/0 | 0.882 | 0.677 | 0.857 | 0.752 | 0.514 | 0.353 |
| 80/20 | **0.884** | 0.702 | 0.859 | 0.789 | 0.581 | 0.402 |
| 60/40 | 0.877 | 0.7 | 0.855 | 0.805 | 0.625 | 0.483 |
| 40/60 | 0.874 | **0.71** | **0.864** | **0.827** | 0.646 | **0.484** |
| 20/80 | 0.846 | 0.664 | 0.839 | 0.793 | **0.651** | 0.481 |
| 0/100 | 0.739 | 0.642 | 0.825 | 0.808 | 0.638 | 0.498 |

[Table 9: F1-score metric for test sets](#D2L_table_label_F1-score metric for test sets)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Recall | | | | | | |
| Model training set Ratio (Normal/Dark) | Test set | | | | | |
| COCO | ExDark | level 1 | level 2 | level 3 | level 4 |
| 100/0 | 0.882 | 0.666 | 0.854 | 0.748 | 0.505 | 0.354 |
| 80/20 | **0.884** | 0.692 | 0.859 | 0.789 | 0.575 | 0.394 |
| 60/40 | 0.876 | 0.69 | 0.853 | 0.803 | 0.617 | 0.474 |
| 40/60 | 0.874 | **0.693** | **0.862** | **0.825** | 0.642 | 0.474 |
| 20/80 | 0.846 | 0.649 | 0.839 | 0.792 | 0.648 | 0.471 |
| 0/100 | 0.744 | 0.631 | 0.824 | 0.806 | **0.637** | **0.492** |

[Table 9: Recall metric for test sets](#D2L_table_label_Recall metric for test sets)

# 

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Precision | | | | | | |
| Model training set Ratio (Normal/Dark) | Test set | | | | | |
| COCO | ExDark | level 1 | level 2 | level 3 | level 4 |
| 100/0 | **0.895** | 0.741 | 0.877 | 0.787 | 0.601 | 0.484 |
| 80/20 | 0.895 | 0.757 | 0.873 | 0.821 | 0.686 | 0.578 |
| 60/40 | 0.891 | 0.755 | 0.869 | 0.822 | 0.676 | 0.579 |
| 40/60 | 0.889 | **0.78** | **0.882** | **0.847** | **0.712** | **0.61** |
| 20/80 | 0.872 | 0.751 | 0.861 | 0.819 | 0.708 | 0.606 |
| 0/100 | 0.784 | 0.722 | 0.847 | 0.829 | 0.679 | 0.569 |

[Table 7: Precision metric for test sets](#D2L_table_label_Precision metric for test sets)

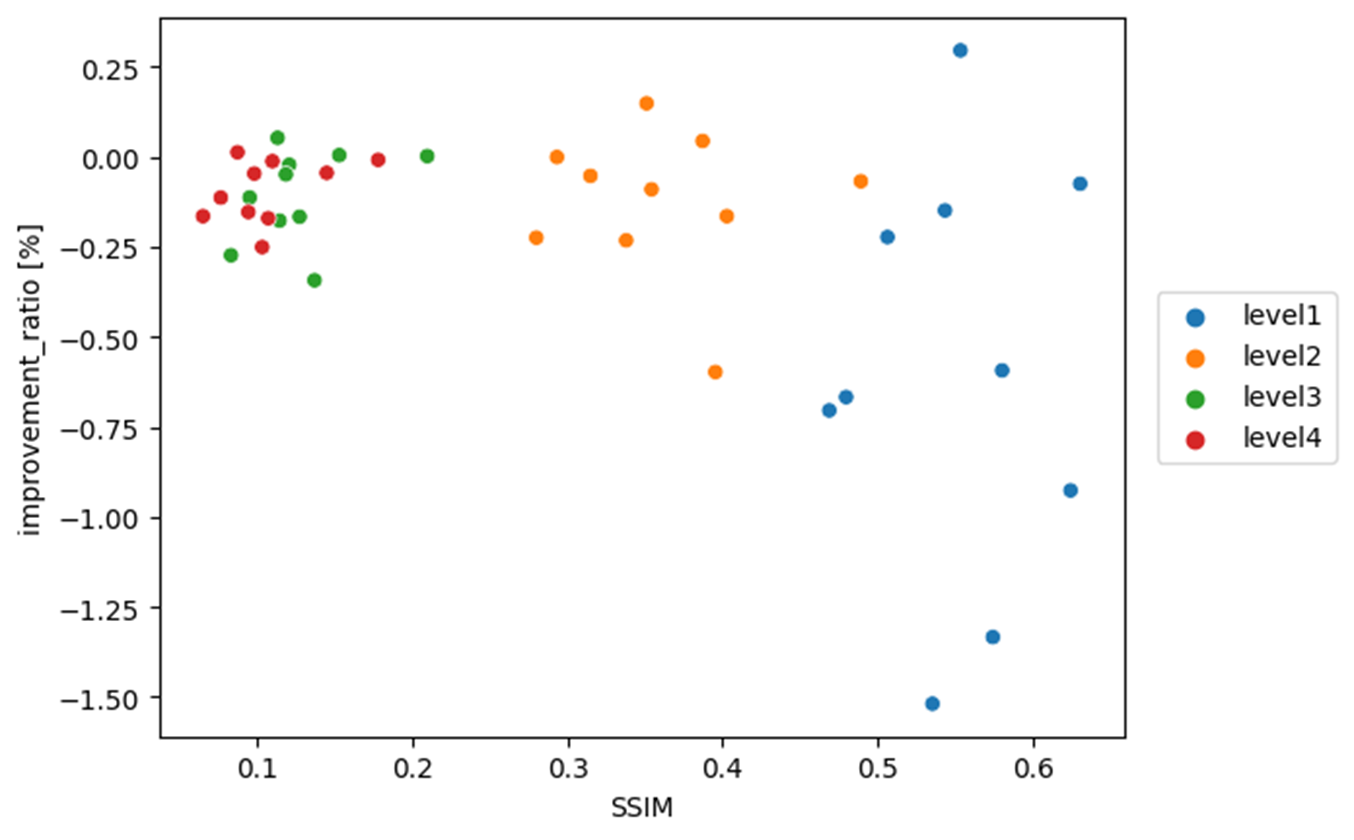
# Summary:

The results indicate that models fine-tuned with fewer low light images have weaker performance in classifying low light datasets, which improves gradually as the ratio of low light images in the training data increases up to 60%. However, pristine dataset performance decreases slightly with fine-tuned models, suggesting a trade-off between the performance on pristine and low-light images. The best overall (pristine and low light) and low light classification model is (Model 40/60), while the best pristine image classification model is (Model 100/0).

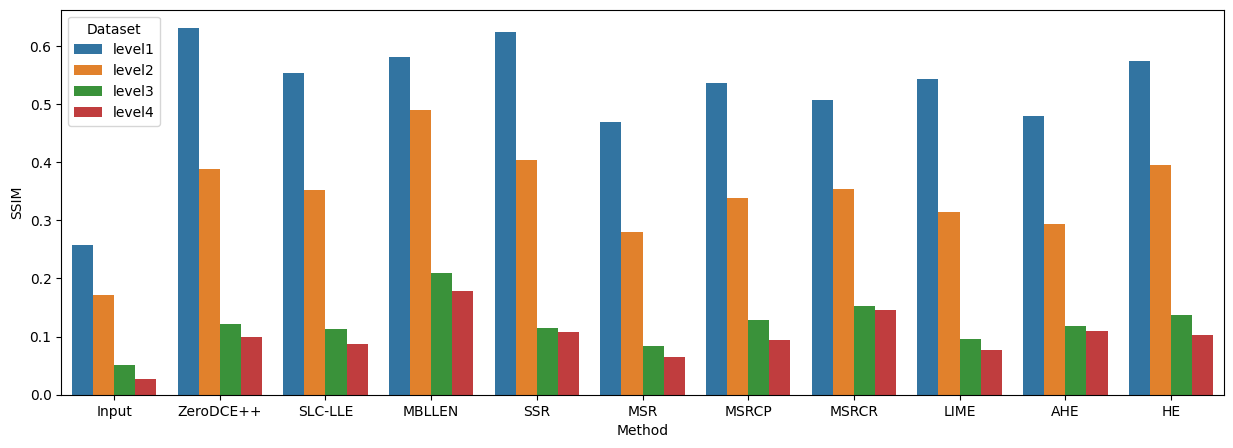
Moreover, we observed that the performance of our model on the ExDark dataset, which consists of real dark images, was not superior to a model trained with real low-light images. This could be attributed to two factors: first, the real images may be better suited to the specific characteristics of the dataset, and second, our augmentation technique may not fully simulate all the complexities of low-light images. Nevertheless, it is worth noting that our approach still yielded a significant improvement in performance.

## 6.5. Image enhancement for classification by pretrained model

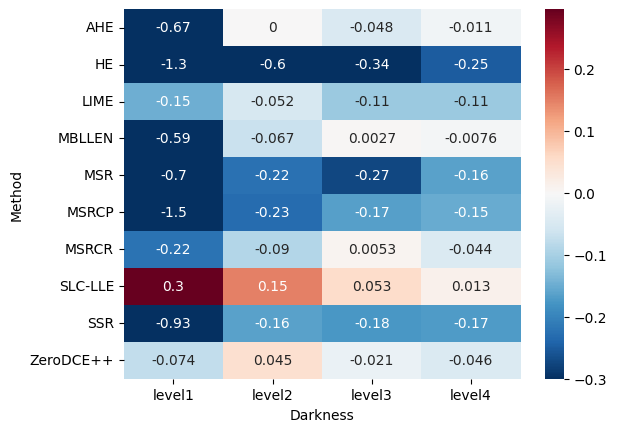
In this experiment, we took the model that was trained only on pristine images and added a preprocessing stage of image enhancement using different methods. From the plot X and fig X (visual compare) we can see that some best enhancement quality is not necessarily to correspond to the best performance. From fig X we can see that the overall best enhancement method is MBLLEN with the highest SSIM score average. However, from fig X we can see that the most of the methods don't improve the classification except for one, SLC-LLE.



The figure show that sometimes best enhancement quality is not necessarily to correspond to the best classification accuracy



SSIM results for different image enhancement methods per darkness level



In Fig XXX showing what methods improve accuracy under different conditions. (X-axis Testset, Y axis - model, hue axis - Improvement ratio)

The tables below are raw results of enhancement methods experiment:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SSIM↑ | level1 | level2 | level3 | level4 |
| Input | 0.258 | 0.172 | 0.051 | 0.027 |
| ZeroDCE++ | **0.630** | 0.387 | 0.121 | 0.098 |
| SLC-LLE | 0.553 | 0.351 | 0.113 | 0.088 |
| MBLLEN | 0.580 | **0.489** | **0.210** | **0.178** |
| SSR | 0.624 | 0.403 | 0.115 | 0.107 |
| MSR | 0.469 | 0.280 | 0.083 | 0.065 |
| MSRCP | 0.535 | 0.338 | 0.128 | 0.095 |
| MSRCR | 0.506 | 0.354 | 0.153 | 0.145 |
| LIME | 0.543 | 0.315 | 0.095 | 0.077 |
| AHE | 0.480 | 0.293 | 0.119 | 0.110 |
| HE | 0.574 | 0.395 | 0.137 | 0.104 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| F1-score | | | | | | |
| Method \ Dataset | COCO | ExDark | level 1 | level 2 | level 3 | level 4 |
| Normal | 0.883 | 0.68 | 0.861 | 0.755 | 0.511 | 0.356 |
| Zero-DCE++ | 0.873 | 0.696 | 0.856 | 0.763 | 0.512 | 0.337 |
| SLC-LLE | 0.885 | **0.703** | **0.866** | **0.775** | **0.539** | **0.371** |
| MBLLEN | 0.875 | 0.669 | 0.842 | 0.745 | 0.511 | 0.35 |
| SSR | 0.863 | 0.673 | 0.834 | 0.735 | 0.458 | 0.276 |
| MSR | 0.876 | 0.663 | 0.841 | 0.725 | 0.42 | 0.278 |
| MSRCP | 0.867 | 0.664 | 0.819 | 0.727 | 0.458 | 0.284 |
| MSRCR | 0.88 | 0.677 | 0.853 | 0.741 | 0.521 | 0.341 |
| LIME | 0.867 | 0.677 | 0.854 | 0.747 | 0.471 | 0.308 |
| AHE | 0.875 | 0.683 | 0.839 | 0.755 | 0.504 | 0.354 |
| HE | **0.886** | 0.667 | 0.823 | 0.682 | 0.382 | 0.212 |

[Table 10: F1-score enhance methods](#D2L_table_label_F1-score ehance methods -d)

Details results for all enhance methods:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| DL Methods | Dataset | acc\_val | Accuracy | Precision | Recall | F1-score |
| Zero-DCE++ | Test | 0.875 | 0.884 | 0.872 | 0.873 | 0.875 |
| ExDark\_test | 0.689 | 0.75 | 0.685 | 0.696 | 0.689 |
| level1 | 0.855 | 0.871 | 0.853 | 0.856 | 0.855 |
| level2 | 0.756 | 0.795 | 0.755 | 0.763 | 0.756 |
| level3 | 0.502 | 0.634 | 0.498 | 0.512 | 0.502 |
| level4 | 0.336 | 0.524 | 0.33 | 0.337 | 0.336 |
| SLC-LLE | Test | 0.885 | 0.895 | 0.883 | 0.885 | 0.885 |
| ExDark\_test | 0.697 | 0.76 | 0.693 | 0.703 | 0.697 |
| level1 | 0.865 | 0.881 | 0.863 | 0.866 | 0.865 |
| level2 | 0.77 | 0.808 | 0.768 | 0.775 | 0.77 |
| level3 | 0.53 | 0.653 | 0.526 | 0.539 | 0.53 |
| level4 | 0.367 | 0.536 | 0.361 | 0.371 | 0.367 |
| MBLLEN | Test | 0.876 | 0.886 | 0.874 | 0.875 | 0.876 |
| ExDark\_test | 0.665 | 0.729 | 0.661 | 0.669 | 0.665 |
| level1 | 0.841 | 0.857 | 0.839 | 0.842 | 0.841 |
| level2 | 0.741 | 0.777 | 0.739 | 0.745 | 0.741 |
| level3 | 0.511 | 0.6 | 0.505 | 0.511 | 0.511 |
| level4 | 0.356 | 0.531 | 0.35 | 0.35 | 0.356 |
| Traditional Methods | Dataset | acc\_val | Accuracy | Precision | Recall | F1-score |
| SSR | Test | 0.864 | 0.878 | 0.861 | 0.863 | 0.864 |
| ExDark\_test | 0.668 | 0.734 | 0.663 | 0.673 | 0.668 |
| level1 | 0.832 | 0.851 | 0.83 | 0.834 | 0.832 |
| level2 | 0.728 | 0.771 | 0.727 | 0.735 | 0.728 |
| level3 | 0.444 | 0.589 | 0.44 | 0.458 | 0.444 |
| level4 | 0.271 | 0.479 | 0.266 | 0.276 | 0.271 |
| MSR | Test | 0.876 | 0.887 | 0.874 | 0.876 | 0.876 |
| ExDark\_test | 0.658 | 0.724 | 0.653 | 0.663 | 0.658 |
| level1 | 0.838 | 0.854 | 0.838 | 0.841 | 0.838 |
| level2 | 0.72 | 0.762 | 0.718 | 0.725 | 0.72 |
| level3 | 0.408 | 0.553 | 0.405 | 0.42 | 0.408 |
| level4 | 0.274 | 0.481 | 0.271 | 0.278 | 0.274 |
| MSRCP | Test | 0.869 | 0.882 | 0.866 | 0.867 | 0.869 |
| ExDark\_test | 0.657 | 0.726 | 0.652 | 0.664 | 0.657 |
| level1 | 0.816 | 0.839 | 0.815 | 0.819 | 0.816 |
| level2 | 0.719 | 0.772 | 0.717 | 0.727 | 0.719 |
| level3 | 0.448 | 0.581 | 0.445 | 0.458 | 0.448 |
| level4 | 0.28 | 0.473 | 0.276 | 0.284 | 0.28 |
| MSRCR | Test | 0.881 | 0.893 | 0.879 | 0.88 | 0.881 |
| ExDark\_test | 0.673 | 0.734 | 0.67 | 0.677 | 0.673 |
| level1 | 0.851 | 0.87 | 0.849 | 0.853 | 0.851 |
| level2 | 0.738 | 0.77 | 0.737 | 0.741 | 0.738 |
| level3 | 0.512 | 0.615 | 0.509 | 0.521 | 0.512 |
| level4 | 0.337 | 0.501 | 0.332 | 0.341 | 0.337 |
| LIME | Test | 0.869 | 0.882 | 0.866 | 0.867 | 0.869 |
| ExDark\_test | 0.671 | 0.732 | 0.665 | 0.677 | 0.671 |
| level1 | 0.853 | 0.868 | 0.851 | 0.854 | 0.853 |
| level2 | 0.743 | 0.783 | 0.742 | 0.747 | 0.743 |
| level3 | 0.468 | 0.595 | 0.465 | 0.471 | 0.468 |
| level4 | 0.301 | 0.498 | 0.297 | 0.308 | 0.301 |
| AHE | Test | 0.877 | 0.885 | 0.874 | 0.875 | 0.877 |
| ExDark\_test | 0.68 | 0.743 | 0.673 | 0.683 | 0.68 |
| level1 | 0.839 | 0.858 | 0.837 | 0.839 | 0.839 |
| level2 | 0.75 | 0.79 | 0.747 | 0.755 | 0.75 |
| level3 | 0.492 | 0.614 | 0.488 | 0.504 | 0.492 |
| level4 | 0.354 | 0.509 | 0.347 | 0.354 | 0.354 |
| HE | Test | 0.887 | 0.897 | 0.885 | 0.886 | 0.887 |
| ExDark\_test | 0.664 | 0.719 | 0.66 | 0.667 | 0.664 |
| level1 | 0.821 | 0.848 | 0.819 | 0.823 | 0.821 |
| level2 | 0.67 | 0.755 | 0.667 | 0.682 | 0.67 |
| level3 | 0.382 | 0.602 | 0.376 | 0.382 | 0.382 |
| level4 | 0.229 | 0.476 | 0.224 | 0.212 | 0.229 |

[Table 11:](#D2L_table_label_) Details results for all enhance methods

## 6.6. Conclusions and discussion

In our research we looked for the best approach for improving image classification for low light images. As of today, the most common practice is to preprocess the images with an image enhancement method before the classifier. First we demonstrated that NN that was trained on normal images provide less than optimal performance when trying to classify low light images (fig XXXX) and this was our baseline model.

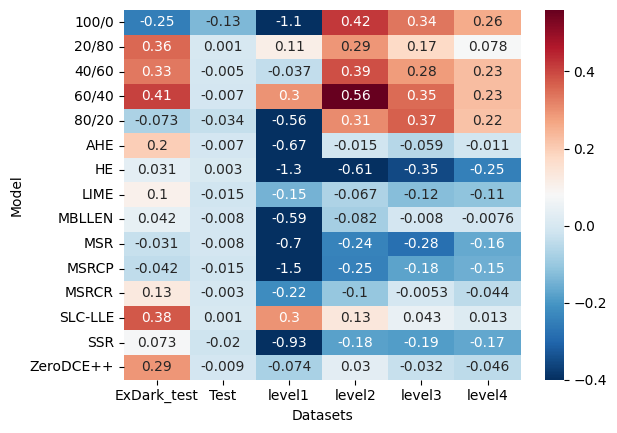
One of the challenges in the field of image classification is creating a big dataset that contains images of the same scenery with different low light conditions. In order to handle this problem we created a synthetic photorealistic method that helps us create low light images of different levels. We chose a very common dataset with normal images and using the photorealistic method we created 4 datasets, each with different low light level. In addition, we included a real dark images dataset in our test set.

We suggested two options for handling our problem.

Experiment 1: in our first experiment we fine tuned our model with datasets that include synthetic dark images along with normal images, with different configurations.

Experiment 2: adding image enhancement before the baseline model.

From our experiments we concluded that even though the most common use method is using image enhancement, it does not yield the best improvement. The best performance improvement was achieved by training the model with a dataset that combines 60% dark images with 40% normal images without very slight degradation of normal image classification.



In Fig XXX showing what methods improve accuracy under different conditions. (X-axis Testset, Y axis - model, hue axis - Improvement ratio)

# 7. Engineering challenges

In this project there are a number of challenges.

First, we will need an environment for running experiments while changing and controlling the various parameters, this environment will need to run deep learning networks and support the following:

* Manage relatively large data sets
* Integrate different computer vision and image process libraries.
* Have the tools for analyzing the results of the tests.

In addition, the environment will need to support and track the different experiments:

* Organizing the different tests and document the results
* Manage and compare the tests results

# 8. Division of work between the partners:

|  |  |  |
| --- | --- | --- |
| **Task** | **Sub task** | **Assignee** |
| **Data augmentation** | Select source image dataset | Together |
| Implement low light image augmentation model | Rom |
| Evaluate the quality of the augmentation | Yarom |
| Generate augment dataset | Rom |
| **Low-light model benchmark** | Evaluate accuracy on pristine images | Rom |
| Estimate accuracy loss as function of degradation parameters | Rom |
| **Image enhancement approach** | Research/Implement image enhancement methods | Yarom |
| Evaluate accuracy improvement as a function of enhancement and degradation parameters | Together |
| **Image augmentation approach** | Design and implement training procedure | Rom |
| Evaluate accuracy improvement over the benchmark as function of degradation and model parameters | Together |
| **Comparative analysis and recommendation** | Compare two approaches for various combinations of parameters | Together |
| Summarize and document recommendations | Together |

[Table 12: Division of work between the partners](#D2L_table_label_Division of work between the partners)

# 

# 

# 9. Required tools

## 9.1. Programs language

We will develop different parts of the projects using matlab and python. We will use the following python libraries:

|  |  |  |
| --- | --- | --- |
| **Tool** | **Description** | **Purpose** |
| Pandas | Open-source library providing high-performance, easy-to-use data structures and data analysis tools. | analysis tools |
| OpenCV | Open-source library that includes several hundreds of computer vision algorithms. | image processing tools |
| TensorFlow | Open-source platform for machine learning. | DP model tools |
| Matplotlib | Matplotlibis a comprehensive library for creating static, animated, and interactive visualizations in Python | analysis tool for visualization |
| Numpy | The fundamental package for scientific computing with Python | Numerical computing tools |

[Table 13: tools](#D2L_table_label_tools)

## 9.2. Develop environments and data analysis:

We will use the following work environments for development and analysis:

|  |  |  |
| --- | --- | --- |
| **Tool** | **Description** | **Purpose** |
| Microsoft Windows 10 | Operating system | work environment |
| GPU | Graphics processor unit | use for DP model calculations |
| python IDE | Code editor | used to write, edit and execute python code |
| Kaggle |  |  |
| Matlab | MATLAB is a programming and numeric computing platform | used to write, edit and execute matlab code |

[Table 14: Develop environments](#D2L_table_label_Develop environments)

# 10. Expected outcomes/deliverables

|  |  |  |
| --- | --- | --- |
| **#** | **Name** | **Description** |
| D1 | Dark image augmentation algorithm | Python library for augmenting dark images from normal images |
| D2 | Augmented Dataset | A data set containing normal and augmented images for training and evaluation |
| D3 | Image enhancement lib | A python library containing different methods for enhancing low light images |
| D4 | Image classification lib | A python library implementation of several image classification pipelines |
| D5 | Image classification framework | A framework for low light image classification |
| D6 | Report and recommendations | An evaluation report and recommendation for selecting the best approach for low light image classification |

[Table 15: Expected outcomes/deliverables](#D2L_table_label_Expected outcomes/deliverables)

# 

# 11. Work plan

|  |  |  |  |
| --- | --- | --- | --- |
| Number | Task name / Title | start date | end date |
| **1** | **Learning** | **28/02/2022** | **17/07/2022** |
| 1.1 | ML course | 28/02/2022 | 30/03/2022 |
| 1.2 | DP courses | 01/05/2022 | 17/07/2022 |
| **2** | **Sow** | **17/04/2022** | **29/06/2022** |
| 2.1 | Literature review | 17/04/2022 | 31/05/2022 |
| 2.2 | Write Sow | 01/05/2022 | 02/06/2022 |
| 2.3 | Submit the SOW to mentor | 29/05/2022 | 29/06/2022 |
| 2.4 | Submission of SOW | 02/06/2022 | 02/06/2022 |
| **3** | **Engineering Report** | **02/06/2022** | **19/01/2023** |
| 3.1 | write engineering report | 01/11/2022 | 01/01/2023 |
| 3.2 | Submission of engineering report | 01/01/2023 | 01/01/2023 |
| 3.3 | Literature review | 02/06/2022 | 30/06/2022 |
| 3.4 | Data augmentation | 15/06/2022 | 20/07/2022 |
| 3.5 | Low-light model benchmark | 21/07/2022 | 18/08/2022 |
| 3.6 | Image augmentation approach | 18/08/2022 | 31/10/2022 |
| 3.7 | Comparative analysis and recommendation (Engineering Report) | 01/01/2023 | 19/01/2023 |
| **4** | **Final Project Report** | **20/12/2022** | **21/05/2023** |
| 4.1 | Image enhancement approach | 20/12/2022 | 15/03/2023 |
| 4.2 | Corrections and improvements | 15/03/2023 | 27/04/2023 |
| 4.3 | Checking results, summary and conclusions | 02/04/2023 | 24/04/2023 |
| 4.4 | Writing a final report | 04/04/2023 | 21/05/2023 |
| 4.5 | Submission of final report | 21/05/2023 | 21/05/2023 |



# 12. References

1. Abdullah-Al-Wadud, Mohammad, et al. "A dynamic histogram equalization for image contrast enhancement." IEEE Transactions on Consumer Electronics 53.2 (2007): 593-600.‏
2. Pizer, Stephen M., et al. "Adaptive histogram equalization and its variations." *Computer vision, graphics, and image processing* 39.3 (1987): 355-368.‏
3. Land, Edwin H. "The retinex theory of color vision." Scientific american 237.6 (1977): 108-129.‏
4. Guo, Xiaojie, Yu Li, and Haibin Ling. "LIME: Low-light image enhancement via illumination map estimation." IEEE Transactions on image processing 26.2 (2016): 982-993.‏
5. Jobson, Daniel J., Zia-ur Rahman, and Glenn A. Woodell. "Properties and performance of a center/surround retinex." IEEE transactions on image processing 6.3 (1997): 451-462.‏
6. Jobson, Daniel J., Zia-ur Rahman, and Glenn A. Woodell. "A multiscale retinex for bridging the gap between color images and the human observation of scenes." IEEE Transactions on Image processing 6.7 (1997): 965-976.‏
7. Llnet: A deep autoencoder approach to natural low-light image enhancement, Pattern Recognition, vol. 61, pp. 650–662, 2017.
8. Lore, Kin Gwn, Adedotun Akintayo, and Soumik Sarkar. "LLNet: A deep autoencoder approach to natural low-light image enhancement." Pattern Recognition 61 (2017): 650-662.‏
9. Wei, Chen, et al. "Deep retinex decomposition for low-light enhancement." arXiv preprint arXiv:1808.04560 (2018).‏
10. He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.‏
11. Bello, Irwan, et al. "Revisiting resnets: Improved training and scaling strategies." Advances in Neural Information Processing Systems 34 (2021).‏
12. Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." International conference on machine learning. PMLR, 2019.‏
13. Russakovsky, Olga, et al. "Imagenet large scale visual recognition challenge." International journal of computer vision 115.3 (2015): 211-252.‏
14. Xiong, Wei, et al. "Unsupervised real-world low-light image enhancement with decoupled networks." arXiv preprint arXiv:2005.02818 (2020).‏
15. Lv, Feifan, Yu Li, and Feng Lu. "Attention guided low-light image enhancement with a large-scale low-light simulation dataset." International Journal of Computer Vision 129.7 (2021): 2175-2193.‏
16. Sakkos, Dimitrios, et al. "Image editing-based data augmentation for illumination-insensitive background subtraction." Journal of Enterprise Information Management (2020).‏
17. Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks." Proceedings of the IEEE international conference on computer vision. 2017.‏
18. Tan, Mingxing, and Quoc Le. "Efficientnetv2: Smaller models and faster training." *International Conference on Machine Learning*. PMLR, 2021.
19. Loh, Yuen Peng, and Chee Seng Chan. "Getting to know low-light images with the exclusively dark dataset." *Computer Vision and Image Understanding* 178 (2019): 30-42.‏‏
20. Lin, Tsung-Yi, et al. "Microsoft coco: Common objects in context." *European conference on computer vision*. Springer, Cham, 2014.‏
21. Xiong, Wei, et al. "Unsupervised real-world low-light image enhancement with decoupled networks." *arXiv preprint arXiv:2005.02818* (2020).‏
22. Agarap, Abien Fred. "Deep learning using rectified linear units (relu)." arXiv preprint arXiv:1803.08375 (2018).‏
23. Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.‏

# 12. Appendices

## 12.1 Code

Git repository: <https://github.com/romhirsch/afeka_project_image_classification_low_light>

### 12.1.1 Interesting functions, classes and scripts

#### 12.1.1.1 Augmentation algorithm

* GammaCorr(img, gamma): applies gamma correction to the image using a lookup table, with a specified gamma value. It returns the corrected image as a float between 0 and 1.
* low\_light\_transform(img, alpha, beta, gamma): applies a low light correction to the image by first applying alpha and gamma correction and then multiplying the result by beta.
* read\_noise(img, var): adds gaussian noise to the image, with a specified variance value.
* blur(img, size=3): applies a Gaussian blur to the image with a specified kernel size.

**def** GammaCorr**(**img**,** gamma**):**

img **=** np**.**clip**(**img**\***255**,** 0**,** 255**).**astype**(**np**.**uint8**)**

lookUpTable **=** np**.**empty**((**1**,** 256**),** np**.**uint8**)**

**for** i **in** **range(**256**):**

lookUpTable**[**0**,** i**]** **=** np**.**clip**(pow(**i **/** 255.0**,** gamma**)** **\*** 255.0**,** 0**,** 255**)**

res **=** cv2**.**LUT**(**img**,** lookUpTable**)**

**return** np**.**float32**(**res**)/**255

**def** low\_light\_transform**(**img**,** alpha**,** beta**,** gamma**):**

**return** beta **\*** GammaCorr**(**alpha **\*** img**,** gamma**)**

**def** read\_noise**(**img**,** var**):**

**return** random\_noise**(**img**,** mode**=**'gaussian'**,** var**=**var**)**

**def** blur**(**img**,** size**=**3**):**

**return** cv2**.**GaussianBlur**(**img**,** **(**size**,** size**),** 0**)**

Function "create\_dark\_images" which takes a list of images and applies our algorithm to create a new set of "dark" images.

**def** create\_dark\_images**(**imgs**,** alphas**=(**0.8**,** 0.9**),** betas**=(**1.5**,** 7**),** gammas**=(**1.5**,** 7**),** var\_noises**=(**0.001**,** 0.01**)):**

dark\_imgs **=** **[]**

**for** img **in** imgs**:**

alpha **=** np**.**random**.**uniform**(**alphas**[**0**],** alphas**[**0**])**

beta **=** np**.**random**.**uniform**(**betas**[**0**],** betas**[**1**])**

gamma **=** np**.**random**.**uniform**(**gammas**[**0**],** gammas**[**1**])**

var\_noise **=** np**.**random**.**uniform**(**var\_noises**[**0**],** var\_noises**[**1**])**

img\_dark **=** low\_light\_transform**(**np**.**float32**(**img**)/**255**,** alpha**,** beta**,** gamma**)**

img\_dark **=** blur**(**img\_dark**)**

img\_dark **=** read\_noise**(**img\_dark**,** var\_noise**)**

dark\_imgs**.**append**(**np**.**clip**(**img\_dark**\***255**,** 0**,** 255**).**astype**(**np**.**uint8**))**

**return** dark\_imgs

The function "preprocess" is designed to be used as part of the preprocessing of image datasets before they are fed into a machine learning model for training. The function takes an parameter p, which is the probability that the image will be transformed to dark images with our augmentation.

**def** preprocess**(**p**=**0.5**):**

**def** \_preprocess**(**x**):**

**if** tf**.**random**.**uniform**([**1**],** 0**,** 1**,** seed**=**seed\_aug**)** **<** p**:**

alpha **=float(**tf**.**random**.**uniform**([**1**],**0.8**,**1**,** seed**=**seed\_aug**)[**0**])**

beta**=float(**tf**.**random**.**uniform**([**1**],** 0.5**,** 1**,** seed**=**seed\_aug**)[**0**])**

gamma**=float(**tf**.**random**.**uniform**([**1**],** 2**,** 7**,** seed**=**seed\_aug**)[**0**])**

var\_noise**=float(**tf**.**random**.**uniform**([**1**],**0.00001**,**0.001**,**seed**=**seed\_aug**)[**0**])**

img\_dark **=** low\_light\_transform**(**np**.**float32**(**x**)** **/** 255**,** alpha**,** beta**,** gamma**)**

img\_dark **=** blur**(**img\_dark**)**

x **=** read\_noise**(**img\_dark**,** var\_noise**)**

x **=** np**.**clip**(**x**\***255**,** 0**,** 255**).**astype**(**np**.**uint8**)**

**return** preprocess\_input**(**x**)**

**return** preprocess\_input**(**x**)**

**return** \_preprocess

#### 

#### 12.1.1.2 Deep learning class

This class provides a framework for training deep learning models, it takes input parameters such as the classes(list of class labels), number of epochs, batch size for training and validation data, image size, number of channels in the images, whether to set all layers as trainable ,only the top classification layers(fineturing) or using pretrain model, learning rate, name of the model, directory to save checkpoints and figures, etc. This class has several methods that are used for setting up and training the model, such as create\_relavant\_folders, pretrain, create\_model ,etc.

**class** **DPhandler(object):**

…

#### 12.1.1.3 DPhandler.py main

The function defines several different test options that can be run, such as:

* Test\_options.RATIO : A test that varies the ratio of dark images in the dataset and evaluates the performance of the model on each ratio.
* Test\_options.ONE\_TEST : A test that evaluates the performance of the model on a single dataset.
* Test\_options.GRID\_SEARCH : A test that performs a grid search to find the best model parameters.
* Test\_options.Enhance : A test that applies enhancement methods to the dark images and evaluate the performance of the model.

#### 12.1.1.4 Dark\_images\_analyze.py and ds\_analyze.py

The scripts "Dark\_images\_analyze.py" and "ds\_analyze.py" calculate the difference between normal and low-light images, and save these feature values in a dataframe. They also provide plots to present the features.

## 12.2 Results appendices

12.2.1 Raw augmentation method results:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Accuracy** | **Precision** | **Recall** | **F1-score** | **bicycle** | **boat** | **bus** | **car** | **cat** | **dog** | **motorcycle** | **person** |
| **Model 100/0** | | | | | | | | | | | | |
| Test | 87.9% | 89.1% | 87.6% | 87.7% | 64.2% | 87.0% | 86.9% | 89.6% | 95.0% | 88.8% | 93.5% | 95.6% |
| ExDark\_test | 69.3% | 75.5% | 69.0% | 70.0% | 65.0% | 85.8% | 64.9% | 93.1% | 70.5% | 58.5% | 56.0% | 57.9% |
| ExDark | 67.2% | 73.9% | 66.9% | 68.0% | 61.3% | 88.1% | 59.6% | 92.6% | 68.3% | 54.3% | 58.3% | 53.0% |
| level1 | 85.6% | 86.9% | 85.3% | 85.5% | 65.8% | 81.7% | 89.1% | 86.4% | 90.0% | 84.3% | 91.1% | 94.1% |
| level2 | 80.4% | 82.2% | 80.3% | 80.5% | 64.2% | 75.6% | 85.4% | 84.0% | 81.4% | 73.1% | 88.7% | 89.6% |
| level3 | 62.0% | 67.6% | 61.7% | 62.5% | 45.0% | 55.7% | 65.7% | 72.0% | 76.4% | 44.0% | 66.1% | 68.9% |
| level4 | 47.9% | 57.9% | 47.4% | 48.3% | 29.2% | 38.9% | 57.7% | 53.6% | 77.1% | 32.8% | 36.3% | 53.3% |
| Model 80/20 | | | | | | | | | | | | |
| Test | 88.5% | 89.5% | 88.4% | 88.4% | 70.8% | 87.8% | 85.4% | 88.0% | 95.0% | 91.0% | 98.4% | 90.4% |
| ExDark\_test | 69.5% | 75.7% | 69.2% | 70.2% | 71.2% | 82.2% | 61.1% | 95.0% | 75.4% | 54.0% | 58.4% | 56.6% |
| ExDark | 68.6% | 74.7% | 68.4% | 69.2% | 67.2% | 87.5% | 58.1% | 93.6% | 72.1% | 51.6% | 60.0% | 57.0% |
| level1 | 86.0% | 87.3% | 85.9% | 85.9% | 67.5% | 79.4% | 87.6% | 88.0% | 91.4% | 86.6% | 96.8% | 89.6% |
| level2 | 79.1% | 82.1% | 78.9% | 78.9% | 58.3% | 64.1% | 86.1% | 90.4% | 88.6% | 66.4% | 91.1% | 85.9% |
| level3 | 57.9% | 68.6% | 57.5% | 58.1% | 40.8% | 35.9% | 66.4% | 61.6% | 92.1% | 32.1% | 59.7% | 71.1% |
| level4 | 40.1% | 57.8% | 39.4% | 40.2% | 20.0% | 24.4% | 48.2% | 41.6% | 87.9% | 19.4% | 29.0% | 44.4% |
| Model 60/40 | | | | | | | | | | | | |
| Test | 87.9% | 89.1% | 87.6% | 87.7% | 64.2% | 87.0% | 86.9% | 89.6% | 95.0% | 88.8% | 93.5% | 95.6% |
| ExDark\_test | 69.3% | 75.5% | 69.0% | 70.0% | 65.0% | 85.8% | 64.9% | 93.1% | 70.5% | 58.5% | 56.0% | 57.9% |
| ExDark | 67.2% | 73.9% | 66.9% | 68.0% | 61.3% | 88.1% | 59.6% | 92.6% | 68.3% | 54.3% | 58.3% | 53.0% |
| level1 | 85.6% | 86.9% | 85.3% | 85.5% | 65.8% | 81.7% | 89.1% | 86.4% | 90.0% | 84.3% | 91.1% | 94.1% |
| level2 | 80.4% | 82.2% | 80.3% | 80.5% | 64.2% | 75.6% | 85.4% | 84.0% | 81.4% | 73.1% | 88.7% | 89.6% |
| level3 | 62.0% | 67.6% | 61.7% | 62.5% | 45.0% | 55.7% | 65.7% | 72.0% | 76.4% | 44.0% | 66.1% | 68.9% |
| level4 | 47.9% | 57.9% | 47.4% | 48.3% | 29.2% | 38.9% | 57.7% | 53.6% | 77.1% | 32.8% | 36.3% | 53.3% |
| Model 40/60 | | | | | | | | | | | | |
| Test | 87.7% | 88.9% | 87.4% | 87.4% | 62.5% | 87.0% | 86.9% | 85.6% | 93.6% | 93.3% | 96.0% | 94.1% |
| ExDark\_test | 70.0% | 78.0% | 69.3% | 71.0% | 63.8% | 86.4% | 60.3% | 93.7% | 69.9% | 66.5% | 56.8% | 57.2% |
| ExDark | 67.4% | 75.3% | 66.9% | 68.3% | 61.3% | 88.7% | 56.2% | 93.3% | 66.8% | 60.0% | 55.3% | 53.5% |
| level1 | 86.5% | 88.2% | 86.2% | 86.4% | 63.3% | 84.0% | 87.6% | 88.0% | 91.4% | 91.8% | 91.9% | 91.9% |
| level2 | 82.7% | 84.7% | 82.5% | 82.7% | 61.7% | 76.3% | 86.1% | 90.4% | 86.4% | 78.4% | 91.9% | 88.9% |
| level3 | 64.5% | 71.2% | 64.2% | 64.6% | 40.8% | 51.1% | 67.2% | 77.6% | 87.9% | 50.0% | 71.0% | 68.1% |
| level4 | 48.0% | 61.0% | 47.4% | 48.4% | 20.8% | 38.9% | 55.5% | 54.4% | 85.0% | 36.6% | 41.1% | 46.7% |
| Model 20/80 | | | | | | | | | | | | |
| Test | 85.0% | 87.2% | 84.6% | 84.6% | 54.2% | 81.7% | 86.1% | 86.4% | 93.6% | 88.1% | 92.7% | 94.1% |
| ExDark\_test | 65.4% | 75.1% | 64.9% | 66.4% | 59.5% | 82.8% | 54.2% | 92.5% | 71.6% | 49.5% | 48.8% | 60.5% |
| ExDark | 62.7% | 72.3% | 62.1% | 63.5% | 55.2% | 83.7% | 48.6% | 90.8% | 68.2% | 49.3% | 48.5% | 52.5% |
| level1 | 84.2% | 86.1% | 83.9% | 83.9% | 55.0% | 82.4% | 89.1% | 85.6% | 91.4% | 82.1% | 91.1% | 94.1% |
| level2 | 79.3% | 81.9% | 79.2% | 79.3% | 56.7% | 74.8% | 80.3% | 87.2% | 85.7% | 71.6% | 90.3% | 86.7% |
| level3 | 65.1% | 70.8% | 64.8% | 65.1% | 41.7% | 55.7% | 67.2% | 80.8% | 88.6% | 46.3% | 69.4% | 68.9% |
| level4 | 47.6% | 60.6% | 47.1% | 48.1% | 25.0% | 42.0% | 52.6% | 59.2% | 85.0% | 32.1% | 37.9% | 43.0% |
| Model 100/0 | | | | | | | | | | | | |
| Test | 75.0% | 78.4% | 74.4% | 73.9% | 36.7% | 62.6% | 81.8% | 84.0% | 92.1% | 70.9% | 83.1% | 84.4% |
| ExDark\_test | 63.7% | 72.2% | 63.1% | 64.2% | 58.9% | 81.7% | 55.7% | 86.2% | 69.9% | 51.5% | 42.4% | 58.6% |
| ExDark | 60.3% | 68.6% | 59.7% | 60.6% | 52.9% | 83.7% | 51.4% | 84.3% | 64.5% | 46.2% | 38.8% | 56.2% |
| level1 | 82.7% | 84.7% | 82.4% | 82.5% | 59.2% | 77.9% | 88.3% | 81.6% | 89.3% | 77.6% | 91.1% | 94.1% |
| level2 | 80.8% | 82.9% | 80.6% | 80.8% | 61.7% | 74.0% | 86.1% | 86.4% | 84.3% | 69.4% | 91.1% | 91.9% |
| level3 | 64.0% | 67.9% | 63.7% | 63.8% | 43.3% | 54.2% | 73.0% | 78.4% | 77.9% | 42.5% | 71.0% | 69.6% |
| level4 | 49.6% | 56.9% | 49.2% | 49.8% | 27.5% | 48.9% | 62.8% | 51.2% | 71.4% | 30.6% | 46.8% | 54.1% |

12.2.2 Raw training with real dark images results:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Accuracy** | **Precision** | **Recall** | **F1-score** | **bicycle** | **boat** | **bus** | **car** | **cat** | **dog** | **motorcycle** | **person** |
| **Model 100/0** | | | | | | | | | | | | |
| Test | 0.824 | 0.839 | 0.822 | 0.822 | 0.633 | 0.771 | 0.774 | 0.872 | 0.929 | 0.806 | 0.887 | 0.904 |
| ExDark\_test | 0.538 | 0.641 | 0.528 | 0.545 | 0.429 | 0.704 | 0.405 | 0.836 | 0.568 | 0.53 | 0.352 | 0.401 |
| level1 | 0.748 | 0.773 | 0.746 | 0.749 | 0.667 | 0.679 | 0.693 | 0.768 | 0.907 | 0.657 | 0.742 | 0.852 |
| level2 | 0.598 | 0.657 | 0.595 | 0.601 | 0.45 | 0.519 | 0.511 | 0.696 | 0.75 | 0.567 | 0.548 | 0.719 |
| level3 | 0.282 | 0.384 | 0.279 | 0.279 | 0.167 | 0.298 | 0.153 | 0.296 | 0.429 | 0.381 | 0.185 | 0.326 |
| level4 | 0.205 | 0.282 | 0.201 | 0.184 | 0.017 | 0.351 | 0.066 | 0.216 | 0.379 | 0.269 | 0.105 | 0.207 |
| Model 80/20 | | | | | | | | | | | | |
| Test | 0.794 | 0.817 | 0.793 | 0.792 | 0.708 | 0.534 | 0.803 | 0.824 | 0.943 | 0.784 | 0.879 | 0.867 |
| ExDark\_test | 0.672 | 0.675 | 0.671 | 0.669 | 0.785 | 0.805 | 0.756 | 0.73 | 0.754 | 0.545 | 0.576 | 0.414 |
| level1 | 0.694 | 0.744 | 0.692 | 0.682 | 0.7 | 0.244 | 0.788 | 0.784 | 0.936 | 0.687 | 0.661 | 0.733 |
| level2 | 0.551 | 0.646 | 0.547 | 0.546 | 0.408 | 0.198 | 0.664 | 0.672 | 0.807 | 0.493 | 0.532 | 0.6 |
| level3 | 0.303 | 0.519 | 0.297 | 0.276 | 0.05 | 0.053 | 0.27 | 0.456 | 0.707 | 0.351 | 0.161 | 0.326 |
| level4 | 0.204 | 0.386 | 0.198 | 0.159 | 0.017 | 0.015 | 0.117 | 0.368 | 0.65 | 0.187 | 0.056 | 0.178 |
| **Model 60/40** | | | | | | | | | | | | |
| Test | 0.83 | 0.839 | 0.829 | 0.83 | 0.758 | 0.71 | 0.861 | 0.864 | 0.857 | 0.866 | 0.839 | 0.874 |
| ExDark\_test | 0.737 | 0.737 | 0.74 | 0.738 | 0.767 | 0.923 | 0.84 | 0.723 | 0.71 | 0.64 | 0.696 | 0.618 |
| level1 | 0.787 | 0.794 | 0.786 | 0.786 | 0.792 | 0.603 | 0.869 | 0.736 | 0.843 | 0.813 | 0.847 | 0.785 |
| level2 | 0.657 | 0.684 | 0.657 | 0.66 | 0.692 | 0.473 | 0.766 | 0.576 | 0.714 | 0.649 | 0.774 | 0.607 |
| level3 | 0.36 | 0.482 | 0.357 | 0.37 | 0.35 | 0.191 | 0.358 | 0.232 | 0.593 | 0.433 | 0.355 | 0.348 |
| level4 | 0.237 | 0.411 | 0.232 | 0.221 | 0.108 | 0.099 | 0.124 | 0.088 | 0.629 | 0.291 | 0.21 | 0.304 |
| **Model 40/60** | | | | | | | | | | | | |
| Test | 0.803 | 0.828 | 0.801 | 0.801 | 0.575 | 0.672 | 0.847 | 0.896 | 0.9 | 0.769 | 0.895 | 0.852 |
| ExDark\_test | 0.734 | 0.738 | 0.735 | 0.734 | 0.767 | 0.929 | 0.847 | 0.717 | 0.689 | 0.635 | 0.608 | 0.691 |
| level1 | 0.759 | 0.774 | 0.757 | 0.759 | 0.675 | 0.626 | 0.869 | 0.64 | 0.857 | 0.784 | 0.847 | 0.756 |
| level2 | 0.609 | 0.653 | 0.607 | 0.617 | 0.567 | 0.511 | 0.723 | 0.432 | 0.714 | 0.627 | 0.718 | 0.563 |
| level3 | 0.309 | 0.481 | 0.304 | 0.306 | 0.358 | 0.176 | 0.299 | 0.12 | 0.7 | 0.336 | 0.121 | 0.319 |
| level4 | 0.206 | 0.44 | 0.2 | 0.179 | 0.2 | 0.061 | 0.124 | 0.056 | 0.743 | 0.172 | 0.048 | 0.193 |
| **Model 20/80** | | | | | | | | | | | | |
| Test | 0.787 | 0.799 | 0.785 | 0.789 | 0.75 | 0.771 | 0.759 | 0.704 | 0.886 | 0.813 | 0.815 | 0.785 |
| ExDark\_test | 0.757 | 0.762 | 0.759 | 0.759 | 0.816 | 0.935 | 0.84 | 0.761 | 0.738 | 0.645 | 0.664 | 0.671 |
| level1 | 0.692 | 0.727 | 0.69 | 0.693 | 0.758 | 0.618 | 0.774 | 0.528 | 0.85 | 0.813 | 0.637 | 0.541 |
| level2 | 0.571 | 0.624 | 0.568 | 0.578 | 0.617 | 0.504 | 0.693 | 0.4 | 0.729 | 0.597 | 0.524 | 0.481 |
| level3 | 0.287 | 0.439 | 0.282 | 0.281 | 0.308 | 0.191 | 0.241 | 0.16 | 0.629 | 0.246 | 0.097 | 0.385 |
| level4 | 0.183 | 0.37 | 0.177 | 0.148 | 0.117 | 0.076 | 0.029 | 0.064 | 0.643 | 0.142 | 0.056 | 0.289 |
| **Model 0/100** | | | | | | | | | | | | |
| Test | 0.658 | 0.688 | 0.656 | 0.644 | 0.608 | 0.573 | 0.803 | 0.28 | 0.743 | 0.769 | 0.935 | 0.533 |
| ExDark\_test | 0.728 | 0.74 | 0.731 | 0.73 | 0.73 | 0.941 | 0.786 | 0.597 | 0.787 | 0.595 | 0.768 | 0.645 |
| level1 | 0.586 | 0.643 | 0.584 | 0.581 | 0.558 | 0.443 | 0.839 | 0.312 | 0.714 | 0.716 | 0.806 | 0.281 |
| level2 | 0.503 | 0.587 | 0.499 | 0.503 | 0.467 | 0.237 | 0.766 | 0.288 | 0.693 | 0.612 | 0.637 | 0.296 |
| level3 | 0.263 | 0.504 | 0.258 | 0.261 | 0.25 | 0.038 | 0.299 | 0.12 | 0.729 | 0.254 | 0.226 | 0.148 |
| level4 | 0.181 | 0.448 | 0.174 | 0.135 | 0.108 | 0.038 | 0.073 | 0.024 | 0.85 | 0.104 | 0.073 | 0.119 |

12.2.2 Raw enhancement method:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Accuracy** | **Precision** | **Recall** | **F1-score** | **bicycle** | **boat** | **bus** | **car** | **cat** | **dog** | **motor** | **person** |
| SLE-LLE | | | | | | | | | | | | |
| Test | 0.885 | 0.895 | 0.883 | 0.885 | 0.708 | 0.885 | 0.905 | 0.928 | 0.893 | 0.933 | 0.895 | 0.919 |
| ExDark\_test | 0.697 | 0.76 | 0.693 | 0.703 | 0.65 | 0.876 | 0.725 | 0.918 | 0.727 | 0.58 | 0.496 | 0.572 |
| level1 | 0.865 | 0.881 | 0.863 | 0.866 | 0.75 | 0.794 | 0.905 | 0.896 | 0.893 | 0.903 | 0.823 | 0.941 |
| level2 | 0.77 | 0.808 | 0.768 | 0.775 | 0.692 | 0.664 | 0.854 | 0.848 | 0.779 | 0.679 | 0.726 | 0.904 |
| level3 | 0.53 | 0.653 | 0.526 | 0.539 | 0.392 | 0.328 | 0.635 | 0.576 | 0.621 | 0.448 | 0.427 | 0.778 |
| level4 | 0.367 | 0.536 | 0.361 | 0.371 | 0.217 | 0.198 | 0.438 | 0.312 | 0.564 | 0.328 | 0.226 | 0.607 |
| MBLLEN | | | | | | | | | | | | |
| Test | 0.876 | 0.886 | 0.874 | 0.875 | 0.675 | 0.885 | 0.883 | 0.92 | 0.886 | 0.91 | 0.895 | 0.933 |
| ExDark\_test | 0.665 | 0.729 | 0.661 | 0.669 | 0.577 | 0.917 | 0.687 | 0.868 | 0.716 | 0.55 | 0.464 | 0.507 |
| level1 | 0.841 | 0.857 | 0.839 | 0.842 | 0.692 | 0.847 | 0.912 | 0.808 | 0.843 | 0.866 | 0.815 | 0.926 |
| level2 | 0.741 | 0.777 | 0.739 | 0.745 | 0.65 | 0.718 | 0.861 | 0.84 | 0.721 | 0.649 | 0.621 | 0.852 |
| level3 | 0.511 | 0.6 | 0.505 | 0.511 | 0.317 | 0.58 | 0.628 | 0.464 | 0.507 | 0.582 | 0.298 | 0.667 |
| level4 | 0.356 | 0.531 | 0.35 | 0.35 | 0.15 | 0.412 | 0.423 | 0.272 | 0.4 | 0.582 | 0.129 | 0.43 |
| SSR | | | | | | | | | | | | |
| Test | 0.864 | 0.878 | 0.861 | 0.863 | 0.65 | 0.87 | 0.869 | 0.936 | 0.9 | 0.925 | 0.831 | 0.911 |
| ExDark\_test | 0.668 | 0.734 | 0.663 | 0.673 | 0.583 | 0.917 | 0.595 | 0.855 | 0.721 | 0.56 | 0.504 | 0.566 |
| level1 | 0.832 | 0.851 | 0.83 | 0.834 | 0.717 | 0.809 | 0.861 | 0.856 | 0.836 | 0.888 | 0.774 | 0.896 |
| level2 | 0.728 | 0.771 | 0.727 | 0.735 | 0.633 | 0.702 | 0.803 | 0.792 | 0.714 | 0.694 | 0.645 | 0.83 |
| level3 | 0.444 | 0.589 | 0.44 | 0.458 | 0.292 | 0.366 | 0.496 | 0.472 | 0.486 | 0.41 | 0.323 | 0.674 |
| level4 | 0.271 | 0.479 | 0.266 | 0.276 | 0.158 | 0.26 | 0.255 | 0.192 | 0.329 | 0.276 | 0.121 | 0.541 |
| MSR | | | | | | | | | | | | |
| Test | 0.876 | 0.887 | 0.874 | 0.876 | 0.708 | 0.855 | 0.905 | 0.936 | 0.886 | 0.903 | 0.887 | 0.911 |
| ExDark\_test | 0.658 | 0.724 | 0.653 | 0.663 | 0.571 | 0.893 | 0.603 | 0.881 | 0.667 | 0.59 | 0.504 | 0.513 |
| level1 | 0.838 | 0.854 | 0.838 | 0.841 | 0.758 | 0.779 | 0.861 | 0.904 | 0.829 | 0.851 | 0.823 | 0.896 |
| level2 | 0.72 | 0.762 | 0.718 | 0.725 | 0.642 | 0.588 | 0.81 | 0.84 | 0.757 | 0.649 | 0.637 | 0.822 |
| level3 | 0.408 | 0.553 | 0.405 | 0.42 | 0.308 | 0.321 | 0.46 | 0.424 | 0.479 | 0.373 | 0.258 | 0.615 |
| level4 | 0.274 | 0.481 | 0.271 | 0.278 | 0.167 | 0.191 | 0.226 | 0.248 | 0.314 | 0.313 | 0.153 | 0.556 |
| MSRCP | | | | | | | | | | | | |
| Test | 0.869 | 0.882 | 0.866 | 0.867 | 0.65 | 0.885 | 0.891 | 0.912 | 0.9 | 0.918 | 0.855 | 0.919 |
| ExDark\_test | 0.657 | 0.726 | 0.652 | 0.664 | 0.558 | 0.893 | 0.664 | 0.855 | 0.65 | 0.59 | 0.464 | 0.539 |
| level1 | 0.816 | 0.839 | 0.815 | 0.819 | 0.725 | 0.771 | 0.883 | 0.848 | 0.779 | 0.881 | 0.742 | 0.889 |
| level2 | 0.719 | 0.772 | 0.717 | 0.727 | 0.617 | 0.634 | 0.839 | 0.816 | 0.636 | 0.739 | 0.613 | 0.844 |
| level3 | 0.448 | 0.581 | 0.445 | 0.458 | 0.292 | 0.344 | 0.518 | 0.552 | 0.436 | 0.403 | 0.298 | 0.719 |
| level4 | 0.28 | 0.473 | 0.276 | 0.284 | 0.142 | 0.237 | 0.285 | 0.264 | 0.336 | 0.261 | 0.137 | 0.548 |
| MSRCR | | | | | | | | | | | | |
| Test | 0.881 | 0.893 | 0.879 | 0.88 | 0.667 | 0.885 | 0.883 | 0.928 | 0.907 | 0.94 | 0.895 | 0.926 |
| ExDark\_test | 0.673 | 0.734 | 0.67 | 0.677 | 0.564 | 0.888 | 0.702 | 0.899 | 0.754 | 0.53 | 0.496 | 0.526 |
| level1 | 0.851 | 0.87 | 0.849 | 0.853 | 0.7 | 0.802 | 0.912 | 0.896 | 0.85 | 0.873 | 0.823 | 0.933 |
| level2 | 0.738 | 0.77 | 0.737 | 0.741 | 0.708 | 0.641 | 0.832 | 0.872 | 0.779 | 0.612 | 0.621 | 0.83 |
| level3 | 0.512 | 0.615 | 0.509 | 0.521 | 0.417 | 0.351 | 0.62 | 0.544 | 0.571 | 0.433 | 0.387 | 0.748 |
| level4 | 0.337 | 0.501 | 0.332 | 0.341 | 0.217 | 0.191 | 0.394 | 0.32 | 0.45 | 0.254 | 0.202 | 0.63 |
| LIME | | | | | | | | | | | | |
| Test | 0.869 | 0.882 | 0.866 | 0.867 | 0.65 | 0.87 | 0.876 | 0.904 | 0.886 | 0.94 | 0.871 | 0.933 |
| ExDark\_test | 0.671 | 0.732 | 0.665 | 0.677 | 0.601 | 0.882 | 0.664 | 0.862 | 0.716 | 0.59 | 0.488 | 0.52 |
| level1 | 0.853 | 0.868 | 0.851 | 0.854 | 0.733 | 0.779 | 0.905 | 0.904 | 0.886 | 0.873 | 0.815 | 0.911 |
| level2 | 0.743 | 0.783 | 0.742 | 0.747 | 0.717 | 0.611 | 0.847 | 0.856 | 0.786 | 0.612 | 0.661 | 0.844 |
| level3 | 0.468 | 0.595 | 0.465 | 0.471 | 0.392 | 0.252 | 0.555 | 0.56 | 0.55 | 0.343 | 0.306 | 0.763 |
| level4 | 0.301 | 0.498 | 0.297 | 0.308 | 0.183 | 0.176 | 0.372 | 0.312 | 0.393 | 0.201 | 0.153 | 0.585 |
| AHE | | | | | | | | | | | | |
| Test | 0.877 | 0.885 | 0.874 | 0.875 | 0.692 | 0.885 | 0.898 | 0.92 | 0.893 | 0.933 | 0.863 | 0.911 |
| ExDark\_test | 0.68 | 0.743 | 0.673 | 0.683 | 0.577 | 0.888 | 0.679 | 0.881 | 0.738 | 0.605 | 0.44 | 0.579 |
| level1 | 0.839 | 0.858 | 0.837 | 0.839 | 0.667 | 0.771 | 0.92 | 0.92 | 0.829 | 0.881 | 0.798 | 0.911 |
| level2 | 0.75 | 0.79 | 0.747 | 0.755 | 0.617 | 0.656 | 0.869 | 0.816 | 0.736 | 0.709 | 0.694 | 0.881 |
| level3 | 0.492 | 0.614 | 0.488 | 0.504 | 0.35 | 0.344 | 0.635 | 0.472 | 0.579 | 0.433 | 0.379 | 0.711 |
| level4 | 0.354 | 0.509 | 0.347 | 0.354 | 0.2 | 0.221 | 0.431 | 0.24 | 0.564 | 0.299 | 0.194 | 0.63 |
| HE | | | | | | | | | | | | |
| Test | 0.887 | 0.897 | 0.885 | 0.886 | 0.667 | 0.901 | 0.891 | 0.92 | 0.893 | 0.948 | 0.927 | 0.933 |
| ExDark\_test | 0.664 | 0.719 | 0.66 | 0.667 | 0.558 | 0.899 | 0.618 | 0.824 | 0.699 | 0.555 | 0.512 | 0.612 |
| level1 | 0.821 | 0.848 | 0.819 | 0.823 | 0.617 | 0.756 | 0.869 | 0.864 | 0.843 | 0.858 | 0.839 | 0.904 |
| level2 | 0.67 | 0.755 | 0.667 | 0.682 | 0.467 | 0.595 | 0.781 | 0.712 | 0.693 | 0.597 | 0.629 | 0.859 |
| level3 | 0.382 | 0.602 | 0.376 | 0.382 | 0.233 | 0.267 | 0.401 | 0.24 | 0.6 | 0.276 | 0.194 | 0.793 |
| level4 | 0.229 | 0.476 | 0.224 | 0.212 | 0.092 | 0.115 | 0.175 | 0.104 | 0.443 | 0.149 | 0.105 | 0.607 |