Computer Vision

Course Project: Efface the haze

Presented by:
Nikhila Dhulipalla, Sasikanth Kotti and Adhun Thalekkara

Presentation Sub-sections

- Introduction
- Implementation details and Results
- Application live demo
- Conclusion
- Limitations

Introduction

- Image dehazing is considered to be a low-level vision task which has got lots of attention in computer vision domain past few decades. Haze, fog, fumes, mist or smoke will greatly reduce the quality of scenery images.
- This leads to image degradation which in turn will lose the contrast and color fidelity.
- So, removal of haze from the images was highly significant to increase the visibility of the scene and also the rectify the color shift caused by the air light.
- With the advance in technology, people are tending to use portable digital devices such as smartphones extensively for capturing images and image reflection is one such issue commonly faced - especially for the images captured through glass or mirror.
- So, it's highly desirable to be removed by using user-friendly image reflection suppression technique which can be used on smart phones within short span of time and obtain a better result in real-time as per the user's visual perception.

Implementation details

The main aim of our project is to apply CV techniques and develop pipeline for image enhancement via Dehazing and removal of image reflections.

We have successfully implemented the below tasks:

1. Implementing the pipeline for image enhancement via Dehazing.
2. Implementing image reflection removal optimization techniques to perform image dehazing.
3. Implementing and reproducing existing results of the paper AAAI 2020 paper - FFA-Net: Feature Fusion Attention Network for Single Image Dehazing.
4. Implementing own architecture to obtain possible improvements of the current/near to SOTA.
5. Apply improved pipeline to video data for dehazing.

Task 1: Implementing the pipeline for image enhancement via dehazing.

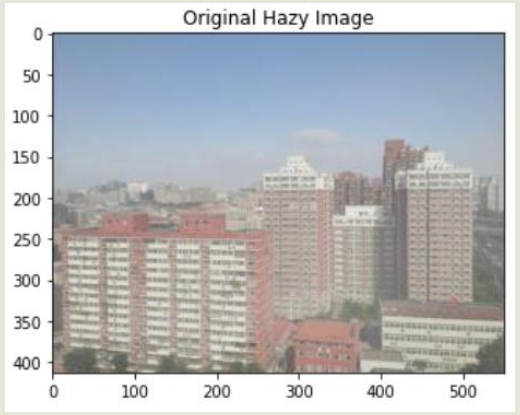
Dark channel prior (DCP) for single image haze removal was implemented and along with this the preprocessing technique used is White Balance (WB) and postprocessing technique used are CLAHE and DWT.

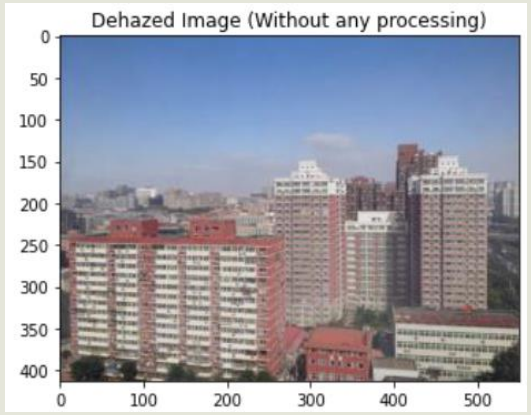
DCP – is a kind of statistics of outdoor haze-free images. Using this prior with the haze imaging model, we can directly estimate the thickness of the haze and recover a high-quality haze-free image.

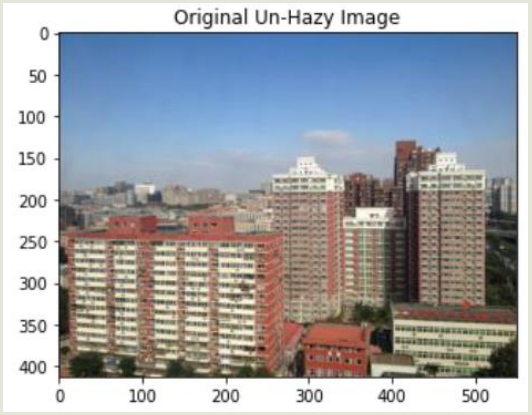
WB – is used as the preprocessing technique to remove the unrealistic color. It helps in balancing the color temperature in the image by adding the opposite color to the image so that the color temperature is neutral.

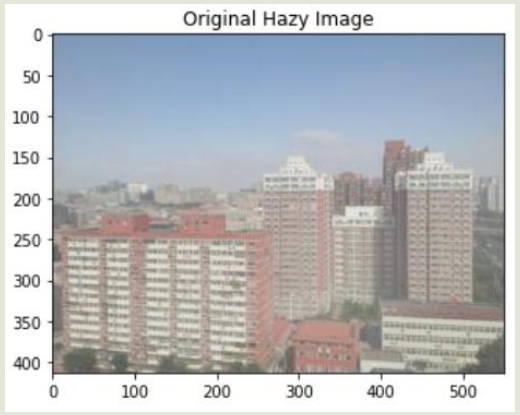
CLAHE – Contrast Limited Adaptive Histogram Equalization is a type of the histogram equalization used as postprocessing technique. It helps in maintaining or limiting the contrast amplification which in turn reduces the noise amplification issue, in simpler words it is used for enhancing the local contrast of an image. Here the vicinity of a given pixel value is given by the slope of the transformation function.

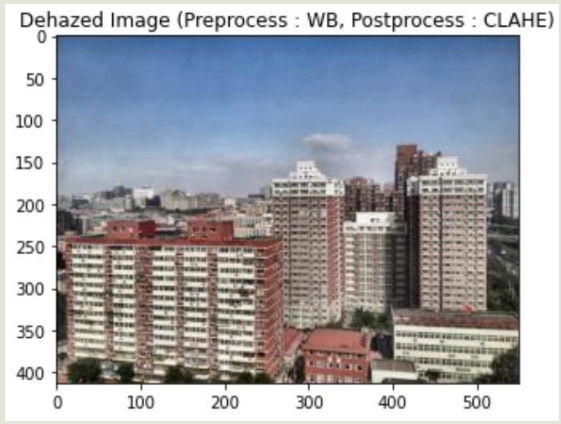
DWT – Discrete wavelet Transform is a technique used for decomposing the signal into multiple sub bands in such a way that low frequency sub bands will be having a finer frequency resolution. This is used as postprocessing technique.

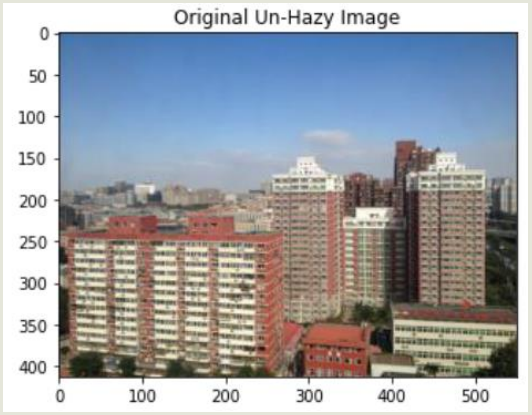

Original Hazy Image

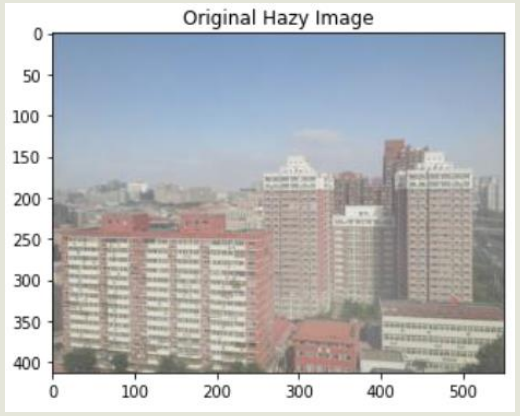

Dehazed Image (Without any processing)

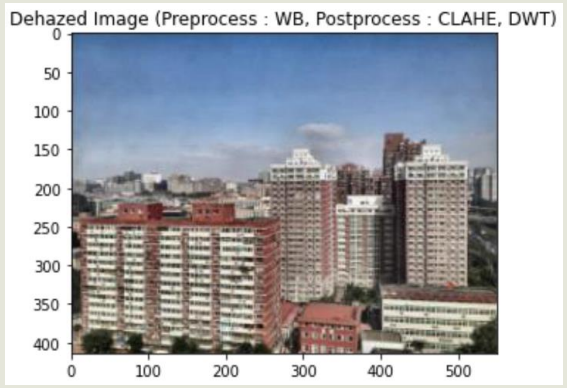

Original Un-Hazy Image

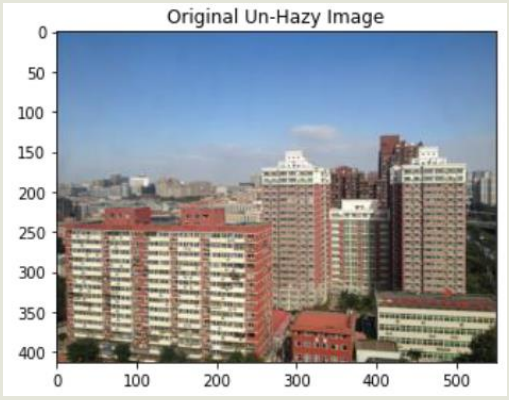

Original Hazy Image


Dehazed Image (Preprocess : WB, Postprocess : CLAHE)


Original Un-Hazy Image


Original Hazy Image


Dehazed Image (Preprocess : WB, Postprocess : CLAHE, DWT)


Original Un-Hazy Image

Implementing the above dehazing technique on the images in the dataset:

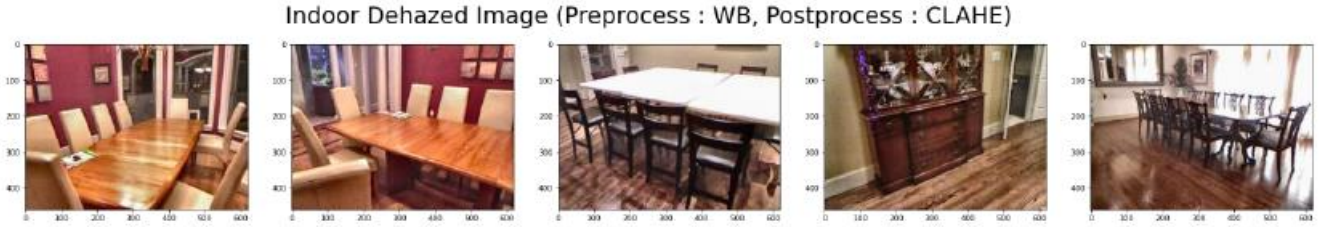
Indoor Hazy images:

Indoor clear images:

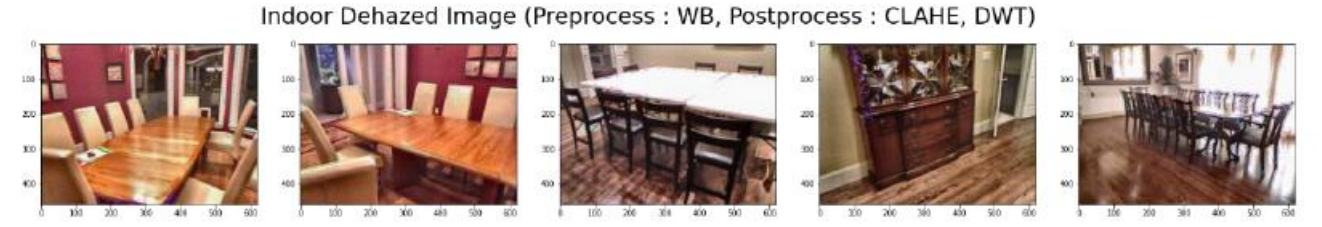
Dehazed images:

Output of DCP on the indoor images:

Output of DCP on the indoor images with preprocessing as WB and postprocessing as CLAHE:

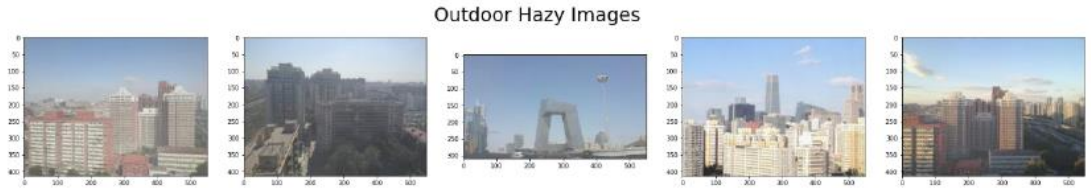


Output of DCP on the indoor images with preprocessing as WB and postprocessing as CLAHE and DWT:

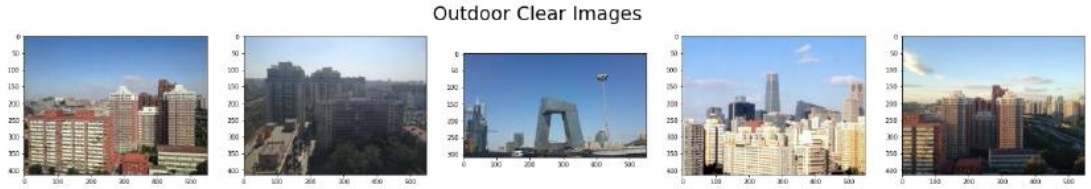


Implementing the above dehazing technique on the images in the dataset:

Outdoor haze images:



Outdoor clear images:



Output of DCP on the outdoor images:

Output of DCP on the outdoor images with preprocessing as WB and postprocessing as CLAHE:

Output of DCP on the outdoor images with preprocessing as WB and postprocessing as CLAHE and DWT:

Evaluation Metrics – PSNR and SSIM

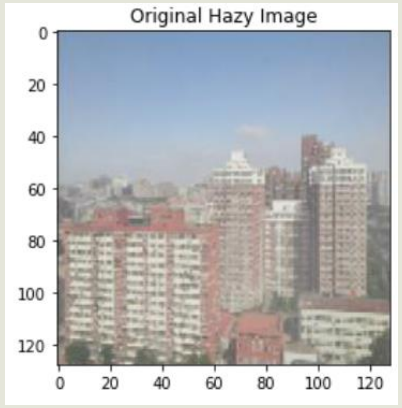
For indoor images:

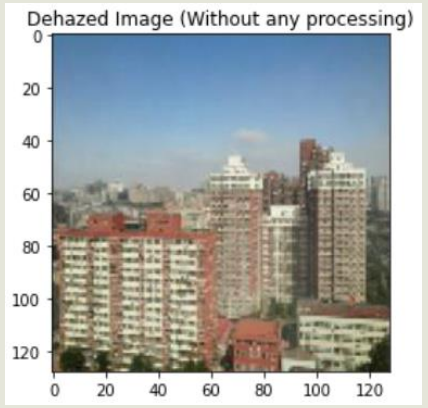
Average PSNR for SOTS Indoor Images Dehazed with DCP	14.9121
Average PSNR for SOTS Indoor Images Dehazed with DCP with Preprocessing(Pipeline1)	**15.0886**
Average PSNR for SOTS Indoor Images Dehazed with DCP with both Preprocessing and Postprocessing (Pipeline2)	14.8982
Average SSIM for SOTS Indoor Images Dehazed with DCP	**0.7808**
Average SSIM for SOTS Indoor Images Dehazed with DCP with Preprocessing(Pipeline1)	0.7487
Average SSIM for SOTS Indoor Images Dehazed with DCP with both Preprocessing and Postprocessing (Pipeline2)	0.7091

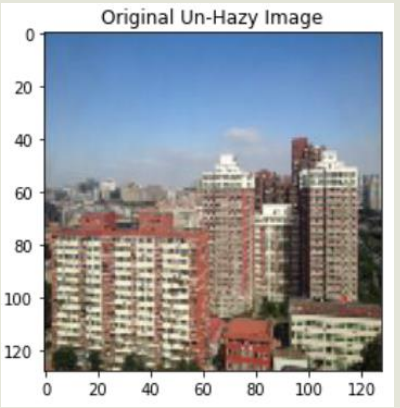
For outdoor images:

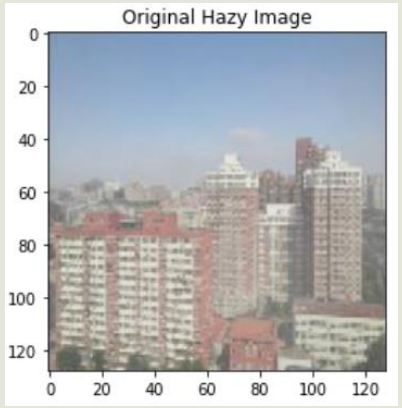
Average PSNR for SOTS Outdoor Images Dehazed with DCP	**21.9988**
Average PSNR for SOTS Outdoor Images Dehazed with DCP with Preprocessing (Pipeline1)	18.5586
Average PSNR for SOTS Outdoor Images Dehazed with DCP with both Preprocessing and Postprocessing (Pipeline2)	18.0762
Average SSIM for SOTS Outdoor Images Dehazed with DCP	**0.9086**
Average SSIM for SOTS Outdoor Images Dehazed with DCP with Preprocessing (Pipeline1)	0.8236
Average SSIM for SOTS Outdoor Images Dehazed with DCP with both Preprocessing and Postprocessing (Pipeline2)	0.7580

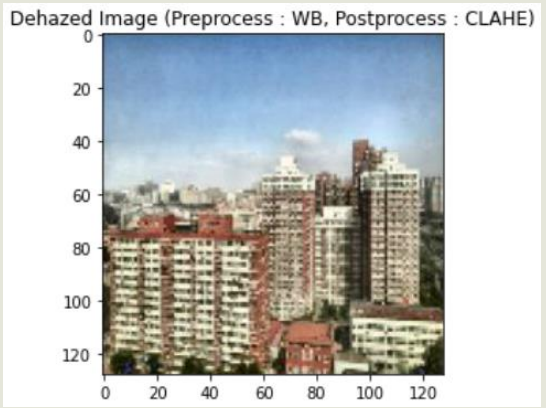
Results obtained on resized image of 128 x 128:

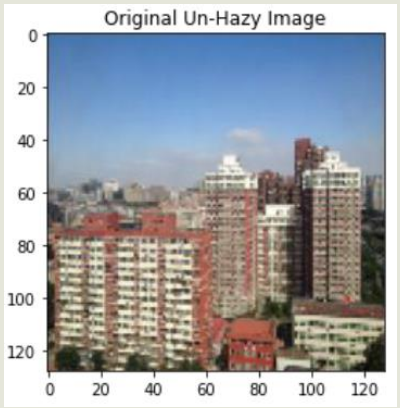


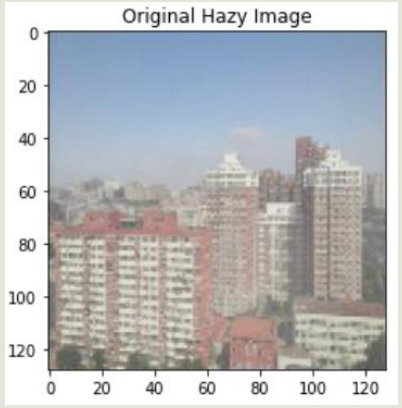


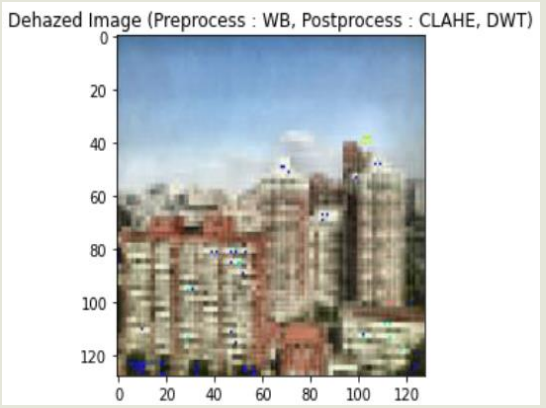


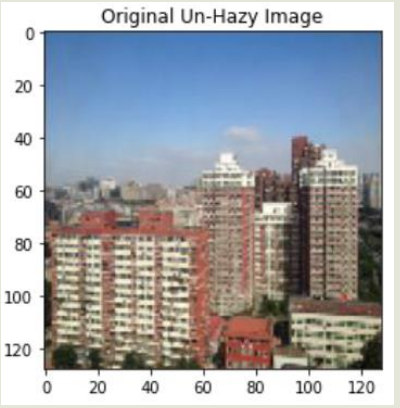












Implementing the above dehazing technique on the resized images of 128 x 128 in the dataset:

Indoor Hazy images:

Indoor Hazy Images

Indoor clear images:

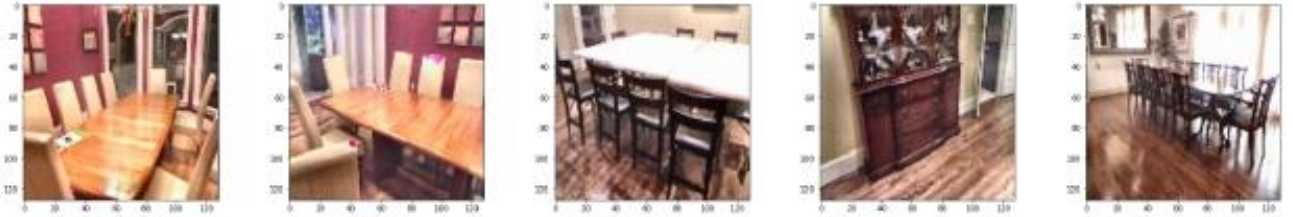
Indoor Clear Images

Output of DCP on the indoor images:

Indoor Dehazed Image (Without any processing)

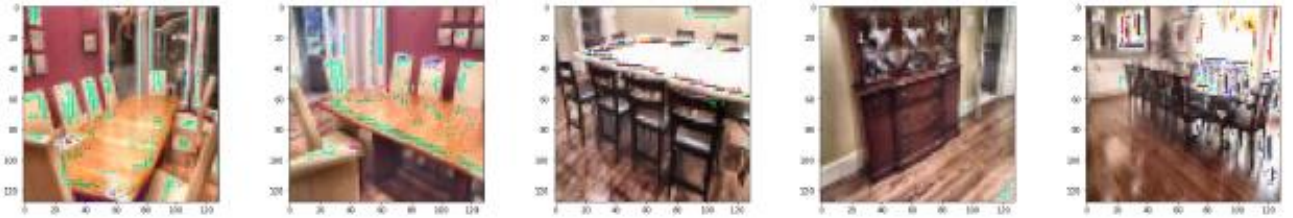
Output of DCP on the indoor images with preprocessing as WB and postprocessing as CLAHE:

Indoor Dehazed Image (Preprocess : WB, Postprocess : CLAHE)



Output of DCP on the indoor images with preprocessing as WB and postprocessing as CLAHE and DWT:

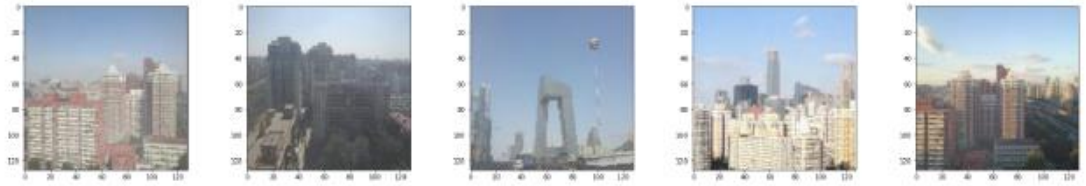
Indoor Dehazed Image (Preprocess : WB, Postprocess : CLAHE, DWT)



Implementing the above dehazing technique on the resized images of 128 x 128 in the dataset:

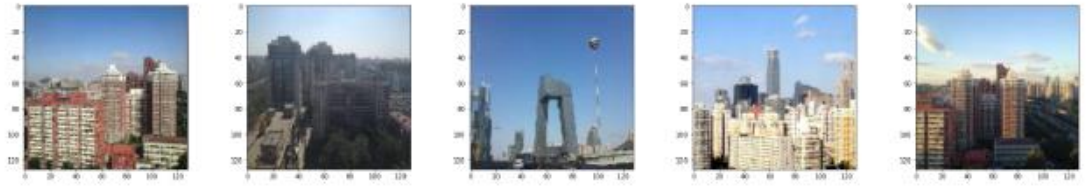
Outdoor haze images:

Outdoor Hazy Images



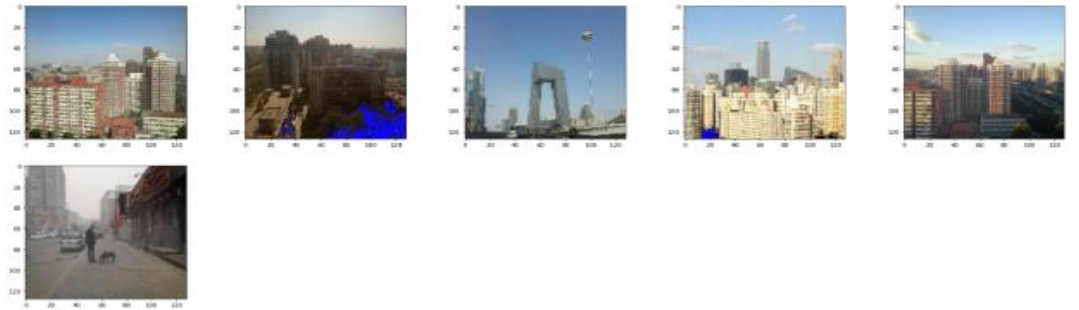
Outdoor clear images:

Outdoor Clear Images



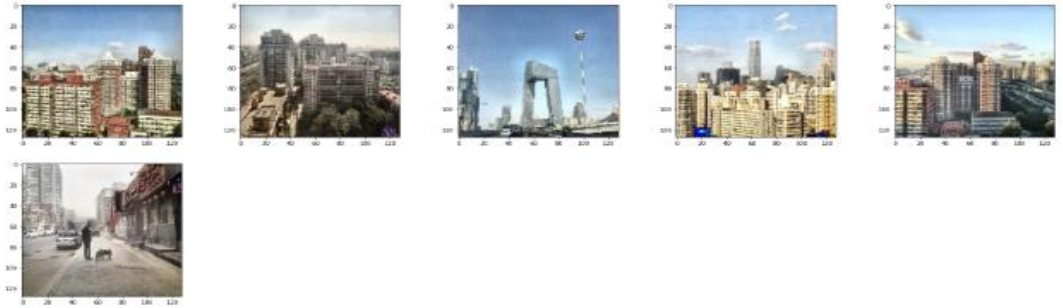
Output of DCP on the outdoor images:

Outdoor Dehazed Image (Without any processing)



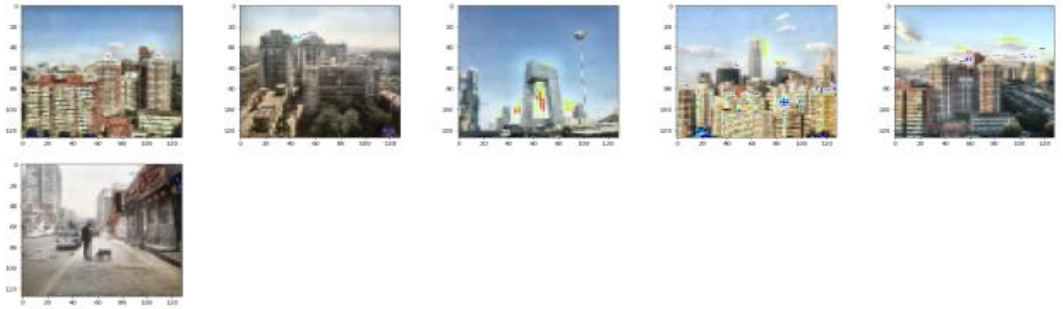
Output of DCP on the outdoor images with preprocessing as WB and postprocessing as CLAHE:

Outdoor Dehazed Image (Preprocess : WB, Postprocess : CLAHE)



Output of DCP on the outdoor images with preprocessing as WB and postprocessing as CLAHE and DWT:

Outdoor Dehazed Image (Preprocess : WB, Postprocess : CLAHE, DWT)



Evaluation Metrics – PSNR and SSIM

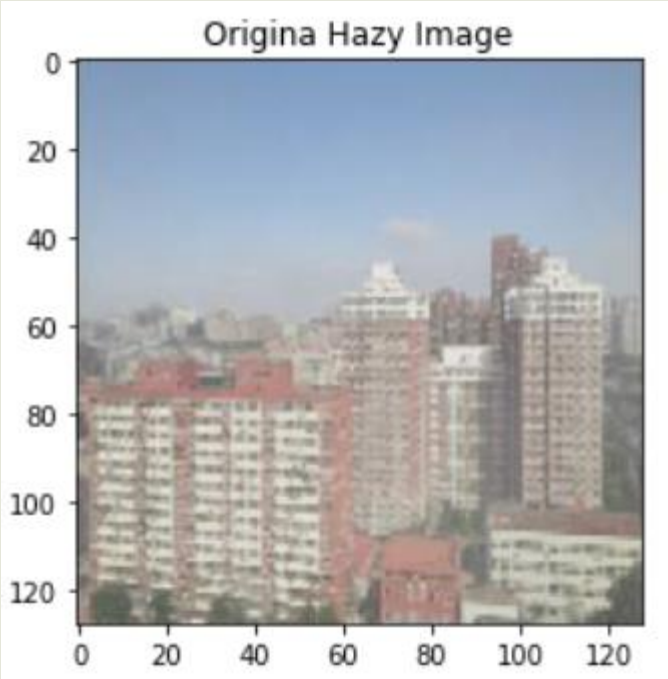
For indoor images:

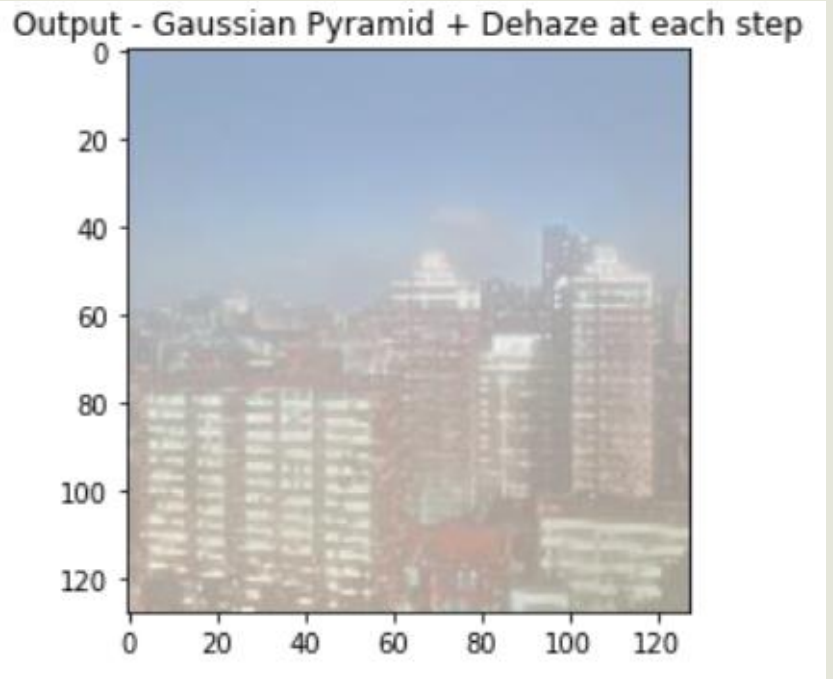
Average PSNR for SOTS Indoor Images Dehazed with DCP	**14.7703**
Average PSNR for SOTS Indoor Images Dehazed with DCP with Preprocessing(Pipeline1)	11.8307
Average PSNR for SOTS Indoor Images Dehazed with DCP with both Preprocessing and Postprocessing (Pipeline2)	11.1531
Average SSIM for SOTS Indoor Images Dehazed with DCP	**0.7756**
Average SSIM for SOTS Indoor Images Dehazed with DCP with Preprocessing(Pipeline1)	0.6779
Average SSIM for SOTS Indoor Images Dehazed with DCP with both Preprocessing and Postprocessing (Pipeline2)	0.5116

For outdoor images:

Average PSNR for SOTS Outdoor Images Dehazed with DCP	**22.6485**
Average PSNR for SOTS Outdoor Images Dehazed with DCP with Preprocessing (Pipeline1)	15.7047
Average PSNR for SOTS Outdoor Images Dehazed with DCP with both Preprocessing and Postprocessing (Pipeline2)	14.9250
Average SSIM for SOTS Outdoor Images Dehazed with DCP	**0.9226**
Average SSIM for SOTS Outdoor Images Dehazed with DCP with Preprocessing (Pipeline1)	0.7712
Average SSIM for SOTS Outdoor Images Dehazed with DCP with both Preprocessing and Postprocessing (Pipeline2)	0.6395

We have tried implementing the Gaussian pyramid but no luck



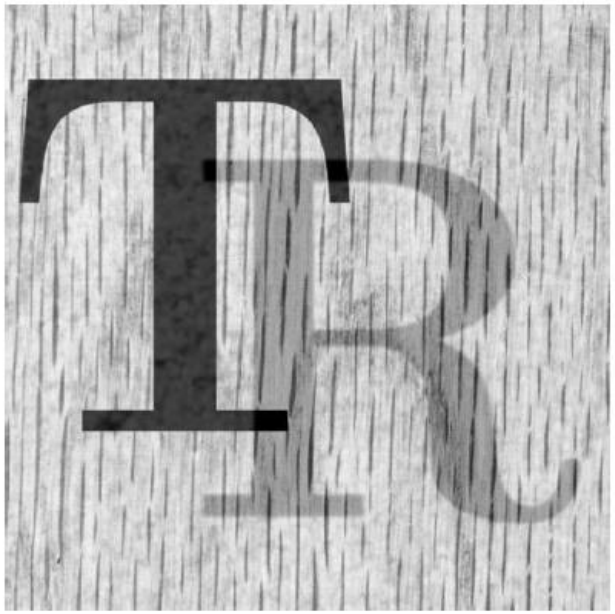


Task 2: Implementing image reflection removal optimization techniques to perform image dehazing.

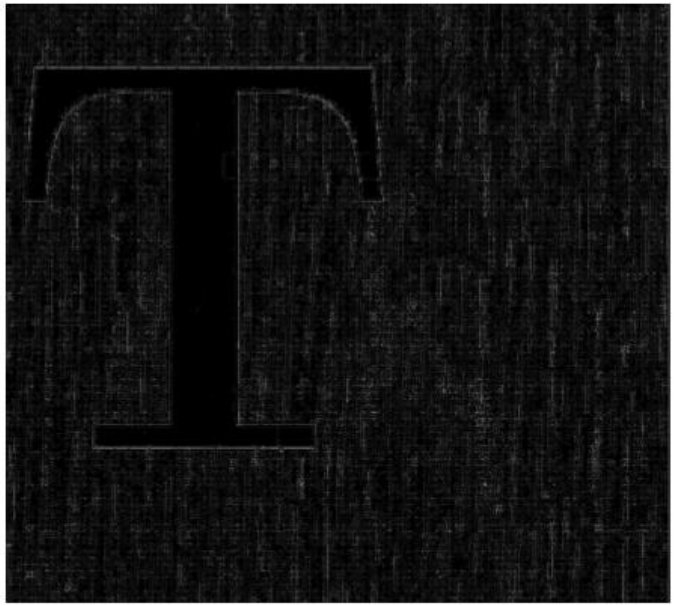
We have implemented the image reflection from one of the efficient approaches proposed in the recent papers, where, the model is convex and the optimal solution is obtained by solving the partial differential equation using Discrete Cosine Transform (DCT)

The base code of paper is on MATLAB and we have tried implementing the same in python but the obtained results are not satisfactory. Hence, we did not apply the reflection removal techniques for image dehazing.

Input image:

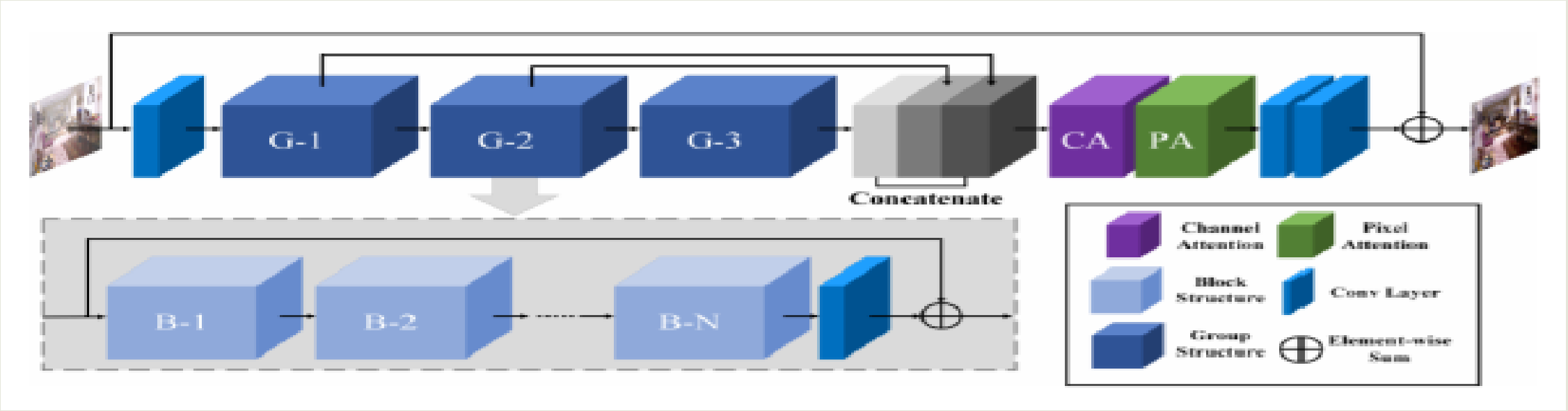


Output image:



Task 3: Implementing and reproducing existing results of the paper AAAI 2020 paper - FFA-Net: Feature Fusion Attention Network for Single Image Dehazing.

We have implemented the FFA-Net for single image dehazing in pytorch.



- FFA architecture was proposed which will help in retaining the shallow layer information by passing it to the deep layers.
- It can fuse all the features and also adaptively learn the different weights at different level feature information, with this it helps in obtaining a better performance.

- We have obtained the inference using pretrained model on resized test images of **128 x 128**.
- We have trained the model from scratch with reduced resolution of **46 x 62**. Below are the results of the both:

Results obtained on pretrained model for the resized images of 128 x 128:

Indoor hazy images:

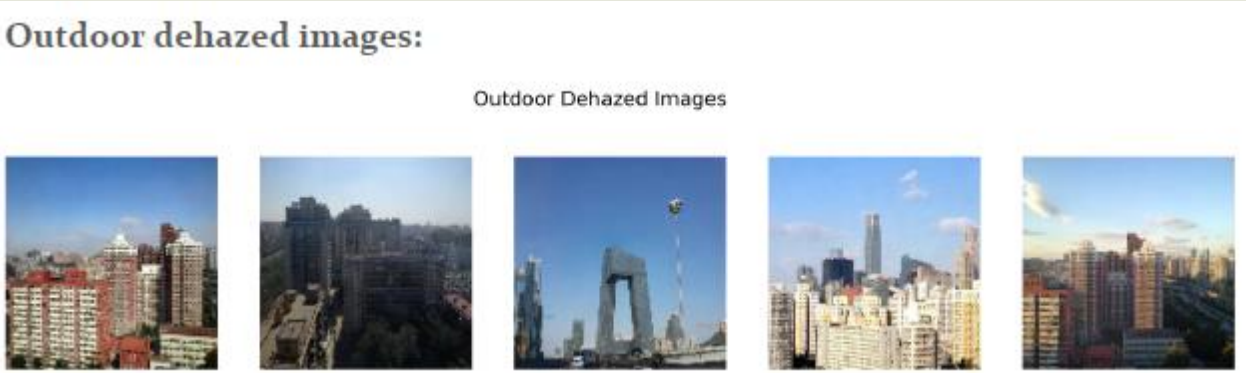
Indoor clear images:

Indoor dehazed images:

Outdoor Hazy images:

Outdoor clear images:

Outdoor dehazed images:



Evaluation Metrics – PSNR and SSIM

For indoor and outdoor images:

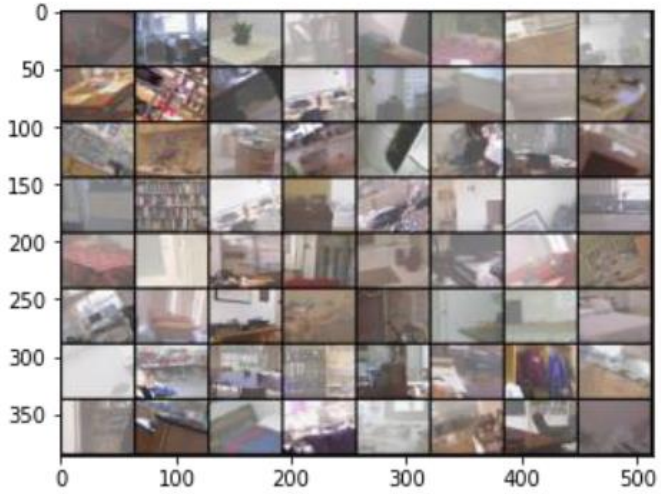
	Test SSIM	Test PSNR
Indoor	0.5559	14.86
Outdoor	0.6363	19.41

Results obtained on model trained from scratch for the resized image of 46 x 62:

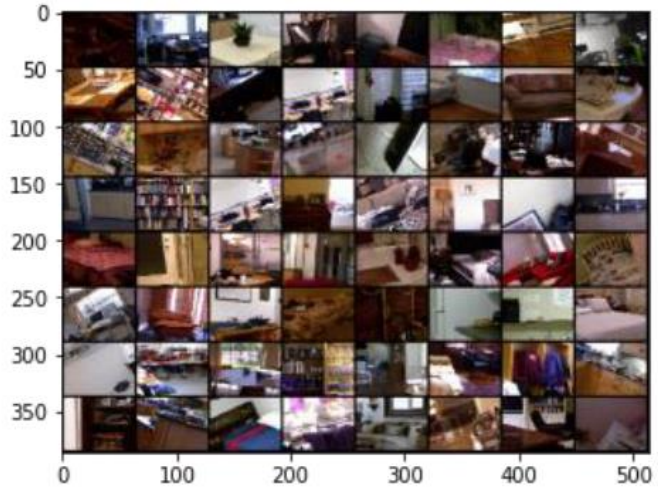
The images are resized to **46 x 62** for the model trained from scratch due to the compute limitations.
Epochs run : 15 – showing the last few steps

epoch 15/15 | step 180/219 | running loss 0.117477560788393 03 | ssim 0.5708408236503602 | psnr 16.39449644088745
epoch 15/15 | step 190/219 | running loss 0.1148621968924999 2 | ssim 0.5787747144699097 | psnr 16.547403621673585
epoch 15/15 | step 200/219 | running loss 0.1187741443514824 | ssim 0.5857049643993377 | psnr 16.25970516204834
epoch 15/15 | step 210/219 | running loss 0.1183858662843704 2 | ssim 0.5773221254348755 | psnr 16.31971321105957
epoch 15/15 | step 219/219 | running loss 0.118756712310844 | ssim 0.5801493724187216 | psnr 16.314223289489746

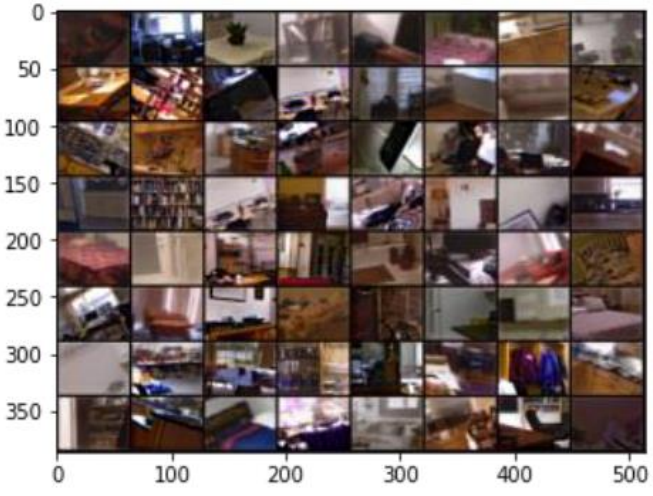
Indoor Hazy image :



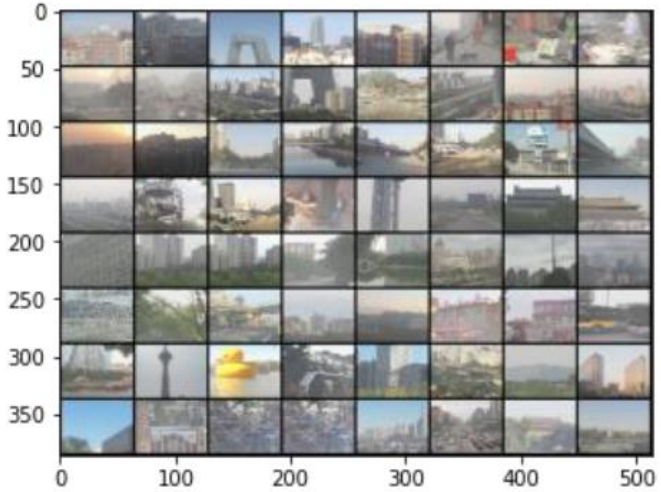
Indoor Clear image:



Indoor Dehazed image:

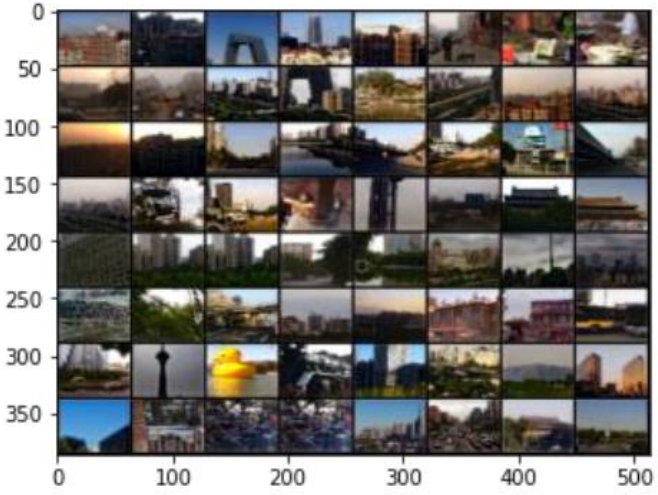


Outdoor Hazy image :



Outdoor Clear image:

Outdoor Dehazed image:



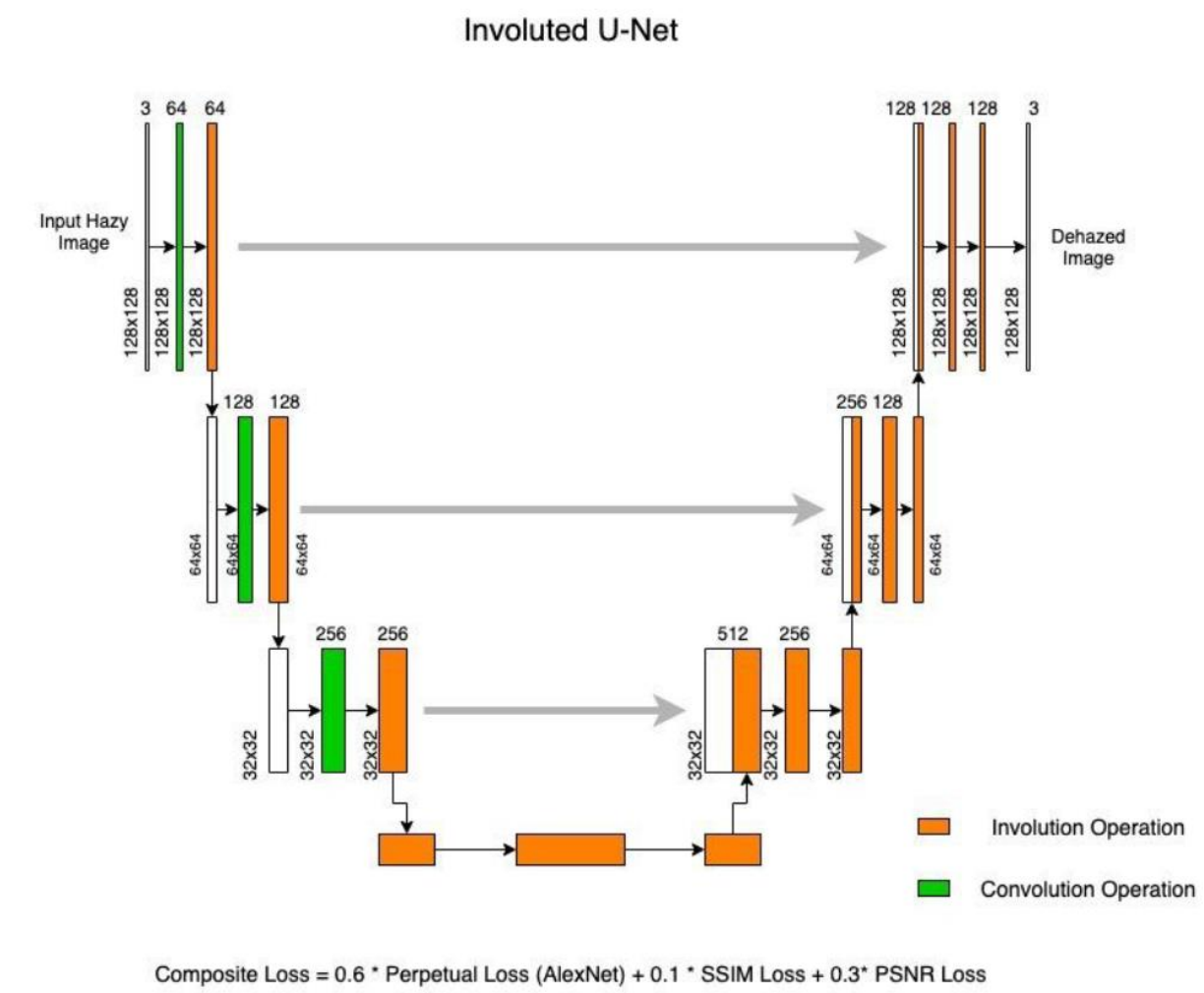
Evaluation Metrics – PSNR and SSIM

For indoor and outdoor images:

	Test SSIM	Test PSNR
Indoor	0.6333	16.73
Outdoor	0.5487	15.99

Task 4: Implementing own architecture to obtain possible improvements of the current/near to SOTA.

Using the U-Net and Involution concepts we have implemented a modified architecture called as **"Involuted U-Net".**



- We have implemented our own architecture to obtain a better performance motivated by U-Net architecture.
- The main difference is that we have used involutions in place of convolutions and applied composite loss - perpetual (Alex net) + SSIM loss + PSNR loss along with weightages.
- The architecture consists of two parts, one is the encoder part (left side in the below architecture diagram) and other is the decoder part(right side in the below architecture diagram).
- The number of channels considered for encoder are 64,128,256 and for the decoder are 256, 128, 64.
- To retain the dimensions of the images, all the involutions and convolutions are padded.
- In the output, instead of generating a single channel mask, were are generating an RGB dehazed image.

Training is carried out for 15 epochs with SGD optimizer, using learning rate 1e-2, weight decay of 0.01 and momentum 0.9 – displaying last two epochs:

========== Epoch: 13

Train loss: -4.3862909356208695, Train SSIM: 0.547225571522293 5, Train PSNR: 15.132052006697638
Test loss SOTS Indoor: -4.532739818096161, Test SSIM SOTS Indoor: 0.585242452144622 8, Test PSNR SOTS Indoor: 15.483044326782226
Test loss SOTS Outdoor: -5.1877481341362, Test SSIM SOTS Outdoor: 0.579035212838552 9, Test PSNR SOTS Outdoor: 17.787031685433735

========== Epoch: 14

Train loss: -4.407835767693715, Train SSIM: 0.556323868279119 6, Train PSNR: 15.192759141792477
Test loss SOTS Indoor: -4.943126082420349, Test SSIM SOTS Indoor: 0.607314200878143 4, Test PSNR SOTS Indoor: 16.847265869140625
Test loss SOTS Outdoor: -4.481691896915436, Test SSIM SOTS Outdoor: 0.550698541528810 4, Test PSNR SOTS Outdoor: 15.506303709696947

Inference on the Best Model

Indoor hazy images:

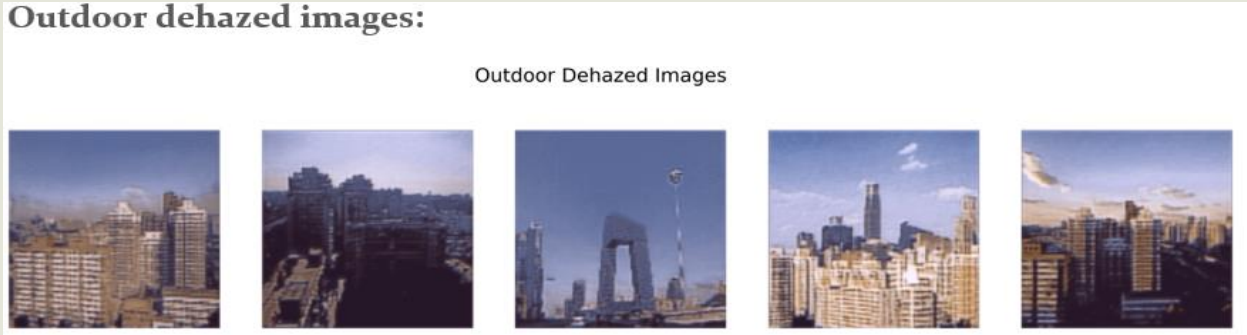
Indoor clear images:

Indoor dehazed images:

Outdoor hazy images:

Outdoor clear images:

Outdoor dehazed images:



Methods	Indoor		Outdoor	
	PSNR	SSIM	PSNR	SSIM
DCP	14.77	0.7757	22.65	0.9226
DCP with Preprocessing and Postprocessing(Pipeline1)	11.83	0.678	15.7	0.7712
DCP with Preprocessing and Postprocessing(Pipeline2)	11.15	0.5116	14.93	0.6395
FFA-Net on pretrained model	14.86	0.5559	19.41	0.6363
Ours	16.85	0.6073	17.78	0.5790

Below are the methods used where the image size are not 128 x 128 so it is not considered for the comparison.

Methods	Indoor		Outdoor	
	PSNR	SSIM	PSNR	SSIM
DCP	14.91	0.7808	21.99	0.9086
DCP with Preprocessing and Postprocessing(Pipeline1)	15.08	0.7487	18.55	0.8236
DCP with Preprocessing and Postprocessing(Pipeline2)	14.89	0.7091	18.07	0.7580
FFA-Net on the model trained from scratch(resized image)	16.73	0.6333	15.99	0.5487

Task 5: Apply improved pipeline to video data for dehazing

We built a video dehazing pipeline using our trained Involuted U-Net and applied it to hazy/foggy video

Hazy video snippet:

Dehazed video snippet:

- As part of viva, we have created a demo version of our implementation and its live on the link – https://24050.gradio.app

- One of the sample is shown below:

Conclusion

- We have shown the results on the pipeline developed for image enhancing via dehazing in task1. We have obtained better results on the indoor images and comparable results on the outdoor images considering the PNSR and SSIM metrics.
- We have implemented and reproduced the results of the FFA-Net paper for both – model trained from scratch and by using pretrained model. For the pretrained model we have resized the image to 128 x 128 and for the model trained from scratch, the images are resized to 46 x 62 due to the GPU limitations (discussion in the next section).
- We have implemented a modified architecture – we have used involutions in place of convolutions and applied composite loss - perpetual (Alex net) + SSIM loss + PSNR loss along with weightages. We have obtained a better result than others.
- The visual quality of the images obtained in Task4 doesn't appear to have the realistic colors. This can be possibly be improved by considering the perpetual loss of VGG.
- Also, if this model is trained on the higher image size or on the original size of the images then we can obtain much better results and visual quality compared to the results obtained FFA-Net paper (possibly by using better GPU's).
- We have implemented all the proposed tasks and we have clearly shown that our own architecture implemented in task4 is giving better results compared to others.

Limitations

The main limitation was with respect to the GPU usage, as we had to run all the codes on colab / colab pro, we had to compromise on things listed below.

- We have resized the images to 128 x 128 due to GPU limitations (in task1).
- We weren't able to run the inference on the test dataset with the actual image size so we had to resize it for both pre-trained and the model trained from scratch (in task3).
- We weren't able to use perpetual loss using VGG due to GPU limitations. So, we have implemented the Alex Net perpetual loss (in task4 and 5).
- We weren't able to replace all the convolutions in the model with involutions. So, we have just replaced for few in the encoder (in task4 and 5).
- We weren't able to add residual connections in both encoder and decoder (in task4 and 5).
- We weren't able to train the model with higher batch sizes.


THANK YOU!