CS230

Report: Deep Learning for Exposure Normalization on Regions of Interest in Digital Images

Vincent Wang Fat Chung
vwfchung@stanford.edu

Rohit Aggarwal
rohitagg@stanford.edu

Benjamin Bay
bay1@stanford.edu

Abstract. Typical HDR image reconstruction requires the use of multiple LDR input images, camera sensor information and complex algorithms with multiple hyperparameters. We present a CNN-based network trained to input a single LDR image and generate a corresponding HDR image with exposure correction. The proposed network consists of two major parts: an LDR encoder and an HDR decoder, with skip connections to allow for efficient exchange of information between the layers. The fully-trained model achieves a PSNR of 15.05 dB, and produces images which clearly demonstrate exposure improvement. Additionally, the network is completely automated, and doesn't require a user to set any parameters. Based on visual evaluation, the model effectively infers important details of regions loss to exposure saturation will help to improve the performance of other classification networks employed as a preprocessing network.

1 Introduction

In the domain of Computer Vision (CV), the problems of exposure normalization and detail preservation for digital images present an important challenge. Optoelectronic image sensors suffer from the problems of under- and over-exposure. Both conditions result in image data loss. While this may be perceived by both humans and machine algorithms performing feature extraction and analysis, this problem is greatest for machines; the human eye is capable of a wider dynamic range than artificial sensors, and is able to rapidly change sensitivity by adjusting the dilation of the pupil.

High dynamic range (HDR) imaging is a modern solution to this problem. It is the compositing and tone-mapping of images to extend the dynamic range beyond the native capability of the capturing device [1]. The conventional non-Deep Learning (DL) approach is to use HDR sensors that interleave multiple captures (typically 4-8) at different exposure settings and reconstruct the scene into a single image using a tone mapping algorithm. These tone mapping algorithms have multiple parameters which need to be tuned on a case by case basis based on subjective human analysis. This is one major drawback of the conventional approach. We propose an alternative solution: to utilize convolutional neural networks (CNNs) to infer the missing color and feature data by mitigating loss of detail in images due to the limited exposure range in a single capture from a single mid-exposure frame.

2 Related work

This area of research is quite specific, and represents much unexplored potential. We build upon the research of two primary groups: Marnerides et al. [2] and Eilertsen et al. [3], both of which published relatively recently. Both groups used CNNs to reconstruct LDR content into HDR. They established that neural nets may indeed approximate the reconstruction of missing information in images.

We do not follow the aforementioned researchers' model architectures exactly, but do draw from their methods, and use their results to inform our own research decisions.

3 Data

Raw image data in Nikon Electronic Format (NEF) files is obtained from the Fairchild HDR Photographic Survey Dataset [4], which provides 1035 images in 105 scenes, each with approximately 9 pictures at different exposures, one f-stop apart. In addition, locally tone-mapped renders are available for each scene; this is a raster image with each of the exposures fused and adjusted to be appealing to subjective human evaluation; these images serve as the ground truths.

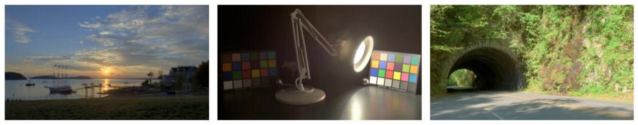

Fig. 1. Three example scenes from the *Fairchild HDR Photographic Survey* dataset [4]. Note the potential high-dynamic range applications, which are manifest as dark regions proximate to bright regions. The human eye is capable of sufficiently wide dynamic range to perceive detail in both regions, while machine sensors fail to do the same in LDR imaging mediums.

As the dataset images are of larger dimensions, the images were further subdivided by scaling and cropping, according to the dimensions of the HDR render. Conventional data augmentation techniques such as image mirroring and rotations in different directions were further employed to augment the limited input data set and improve variance metrics. Altogether, these techniques provide 14 sample pairs per LDR exposure of a scene.

The augmented dataset is split into train, development and test sets in the ratio of 80/10/10 (Table 1. Since the dataset contains multiple captures per scene (at different exposures), the split is done on a per scene basis to ensure that the development and test datasets are previously unseen by the model.

Table 1. Dataset augmentation and splitting, no. of samples.

Dataset split		
Train	Dev	Test
10318	1260	1260

During training, a serious misalignment issue was discovered between input and output pairs. Due to cropping and scaling performed on the HDR renders, the field of view became different from the input raw files. Data augmentation, which employed rotations and flips, further exacerbated this issue. The resultant original model produced blurry and misaligned images with heavy artifacts. It was resolved by adding an image registration stage before data augmentation, using the *Enhanced Correlation Coefficient Maximization*[5] algorithm in the implementation provided by OpenCV.

4 Model

Our CNN architecture includes convolutional, pooling, and deconvolutional layers.

4.1 Loss

Our primary loss function is L1 loss (or mean absolute error). We also included L2 loss (or mean square error) since that helped improve PSNR. Moreover, to help reduce noise, **total variation** (TV) loss was also added. This is used to reduce the gradient between adjacent pixels, which has the effect of de-noising the image. We selected a linear combination of L1 and TV loss as our final loss function.

$$\begin{aligned}\mathcal{L}_{custom}(\hat{y}) = \sum_{i,j}&|y-\hat{y}|+ \\ &\sum_{i,j}((y_{i,j+1}-y_{i,j})^2+(y_{i+1,j}-y_{i,j})^2)^{\beta/2},\end{aligned}$$

with overall cost being calculated thus:

$$\mathcal{J}(w,b) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}_{custom}(\hat{y}_i, y_i) \tag{1}$$

4.2 Topology

A diagram of our architecture is presented in Figure 2. We chose to use an autoencoder because of its ability to transform high-dimensional input to a lower-dimensional latent representation, and likewise the decoder was used to reconstruct the full-dimensional data [6]. More specifically, our autoencoder has the role of "denoising" [7]; that is, it is trained with corrupt input images (LDR images with exposure data loss), with the goal of extracting important features, and reconstructing original HDR images with complete information.

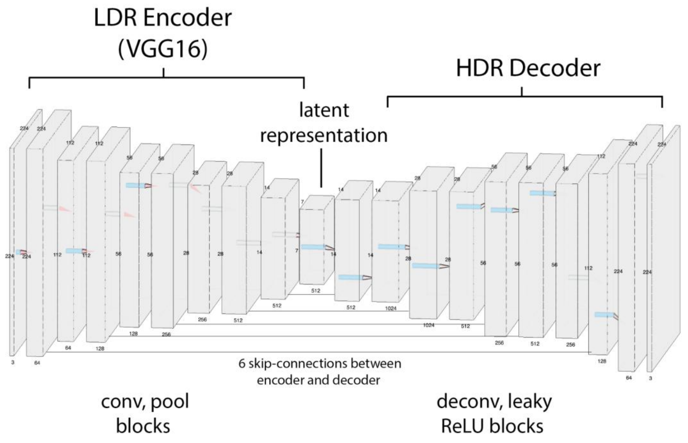
LDR Encoder (VGG16)
latent representation
HDR Decoder
6 skip-connections between encoder and decoder
conv, pool blocks
deconv, leaky ReLU blocks

Fig. 2. Neural network model architecture diagram for exposure normalization in digital images. This final model featured 22,467,992 total parameters, with 58.96% trainable and 41.04% non-trainable. The LDR encoder used VGG-16, which requires a series of convolutional layers and max-pool layers. The $7\times7\times512$ latent representation then passes through the HDR decoder, which uses deconvolution (upsampling) layers in combination with leaky ReLU ($\alpha = 0.3$) and skip-connection points.

The encoder is comprised of the first five 5 of the VGG-16 network with the fully-connected layers removed. This is followed by a $7 \times 7 \times 512$ convolution layer, the smallest in the network, which acts a compressor for the latent representation code. The decoder comprises 6 stacked convolution and deconvolution (upsampling) layers, doubling the dimensions to go from the compressed latent representation back into the full $224 \times 224 \times 3$ HDR RGB image output.

The final architecture uses 6 skip connections, to feed the features back to decoder to help image reconstruction. The intuition behind multiple skip connections is to provide the decoder with both low level features from initial layers and high level features from deeper layers. Finally, input image is also fed back and used for reconstruction. This was done, since most of the high resolution detail is lost during feature extraction which might be useful for image reconstruction. All the connections were added before the max pooling layer of decoder to avoid any loss in data. We experimented with models consisting of different number of skip connections to converge on the most optimal architecture for our dataset.

The input and output images are normalized to be between 0 and 1, each of the three 8-bit channels are divided by the maximum value of 255. This makes the sigmoid functi noespecially suitable for use as the activation for the output layer, since their ranges coincide.

We experimented with both ReLU and leaky ReLU ($\alpha = 0.4$) activations for all our convolution and deconvolution layers. For the final output layer activation, we experimented with both ReLU and sigmoid activation. Since ReLU allowed values above 1, this leads to saturation and color shift in output images, and hence the final architecture uses sigmoid activation.

4.3 Training Strategy and Hyperparameter Tuning

In order to augment our model's comprehension of digital images, we initialized our encoder section of the model with pre-trained VGG-16 weights [8]. This practice is commonplace in the CV research community. Initially, we froze all VGG-16 layers from training. The two additional layers added after VGG-16 were added and trained to help refine the features with respect to our dataset. All the added deconvolution and skip connection layers were initialized using the default Glorot uniform initializer in Keras [9] and trained on our dataset.

Since our network outputs image, any expected loss function should look at pixel-wise comparison and use the sum over all pixels and on all examples in a mini-batch. We considered L2 loss as the first choice, as it directly helped improve our evaluation metric of PSNR. However, we noticed that L1 loss produced sharper images compared to L2 loss, but it had more checkerboard like artifacts. Total variation loss helped to make the images smoother and reduce these artifacts, by reducing the gradient of pixels across rows and columns.

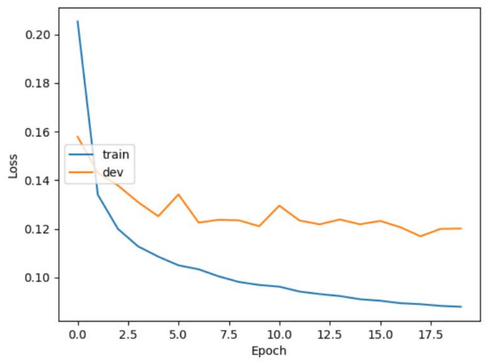

Fig. 3. Train and development set loss over 20 epochs of model training. We employed a combination of total variation and L1 loss

We also noticed that our model was overfitting the training set due to the lack of training examples. We experimented with dropout and L2 regularization of bias and kernel in all our convolutional layers. This helped reduce overfitting, however, this led to the network not being able to fit training set well. To improve upon this, we set some layers of VGG-16 encoder to be trainable. The approach to setting this was slightly different compared to typical transfer learning. Instead of allowing the last few layers to be trainable, we set the layers with a direct skip connection to the decoder as trainable. This approach is intuitive for our application, since refining the layers feeding directly to the decoder should give more immediate improvement. We tried models with other variations as well, however, allowing more layers to train degraded results probably due to lack of enough training data.

4.4 Evaluation Metric

Peak Signal to Noise Ratio (PSNR) is an established metric in CV, used to quantitatively compare different encodings of the same image [10]; we implemented this function as the evaluation metric for our Keras model. OpenCV's built-in HDR tone-mapping functions were used to generate the baseline metrics; all LDR captures along with their exposure times were used to compute the camera response function using Debevec's weighting scheme [11], and were then mapped and fused into an 8 BPP RGB representation using Reinhard's algorithm [12]. These images, compared against the locally rendered HDR images using PSNR, form the baseline metric.

$$PSNR(im1, im2) = 20 \log \frac{M}{RMSE(im1, im2)}, \tag{2}$$

where M is the max pixel value. RMSE is defined as:

$$RMSE(im1, im2) = \sqrt{\frac{\sum_{\forall N} ||x_{im1}||^2 - ||x_{im2}||^2}{N}}, \tag{3}$$

where N is the number of pixels.

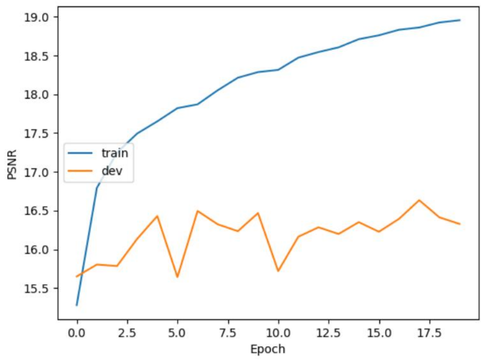

Fig. 4. Train and development set PSNR over 20 epochs of model training, as given by equation 2

Apart from PSNR, visual inspection plays a very important role in evaluating image reconstruction. Importantly, PSNR declines sharply in regions of movement with abrupt changes in contrast, since it compares pixel-to-pixel difference in color channel intensity. Since there were only around 100 scenes, it was reasonable to visually inspect random samples and known scenes for qualitative evaluation.

The **structural similarity** index (SSIM) measures the similarity between two images, where $SSIM : (x, y) \rightarrow$

$[-1, 1]$ with x and y being the windows of their corresponding image (including average, variance, and covariance) [13]. We implemented the SSIM in our code; while it did approach 1 during training we deemed PSNR to be a more useful metric for analysis as it had more correlation with visual quality of the image. For this reason we omit the detailed analysis for SSIM.

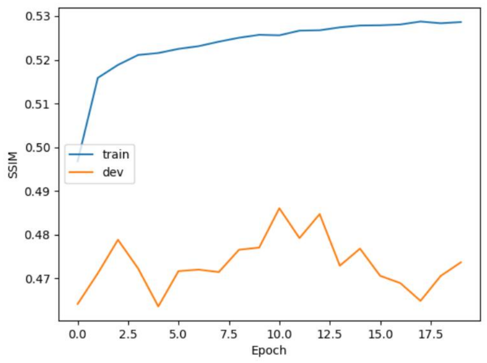

Fig. 5. Train and development set SSIM (structural similarity) over 20 epochs of model training. Note that as training advances, our SSIM does approach 1, albiet slowly.

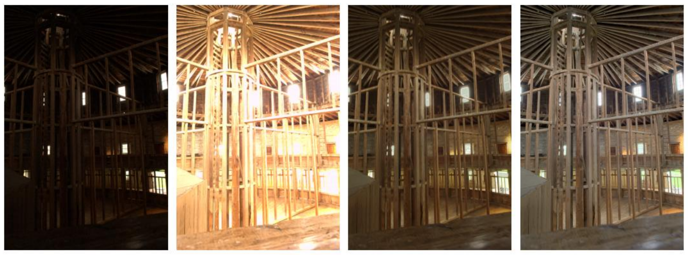

Fig. 6. Left to right: an underexposed LDR image, an overexposed LDR image, an image generated with OpenCV's baseline method, and the ground truth rendered version. Note that this final version is the most photorealistic to human observers.

5 Experiments/Results/Discussion

Our final tuned convolutional autoencoder neural network achieved a PSNR of 15.05 dB on the test dataset (new scenes reserved from training and development), less than the multi-exposure OpenCV tone-mapped baseline of 27.81 dB (see Table 2).

Table 2. Dataset PSNR metrics for baseline (OpenCV tonemapping) and final neural network (Convolutional Autoencoder) models.

Model	Dataset PSNR [dB]		
	Train	Dev	Test
Baseline (OpenCV)	27.90	27.80	27.81
Neural Network	16.35	16.10	15.05

Compared with the baseline OpenCV renders, the convolutional autoencoder network exhibits high bias (>10 dB), but comparatively low variation between the train, dev and test datasets (this is true before the train dataset begins to overfit and validation PSNR saturates at approximately 16.5 dB most model iterations). L2, L1, and dropout regularization were experimentally introduced to attempt to raise the test dataset PSNR; while these raised PSNR to a maximum of 16.86 dB and reduced variance, they did not manage to penetrate 17.0 dB.

While withholding the dataset size (1,000 or 5,000 or 10,000 example pairs), the PSNR converges more slowly and saturates at a lower level than with the full set. This suggests that other factors, such as the limited training examples and lack scene diversity, may play a greater role than the actual network architecture or its hyperparameter tunings.

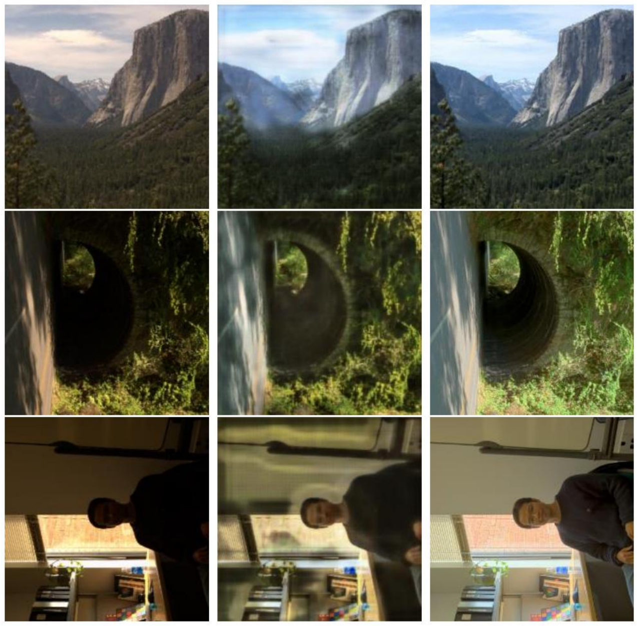

Fig. 7. Development dataset examples through the final model. Left to right: Input (x), Reconstruction ($\hat{y}$), and Ground Truth (y).

The network generates notably different images from the input, inferring higher light intensities and colours for regions such large contiguous regions such as the ground and sky. Under- and overexposed images appear more balanced in terms of colour and brightness, in this respect bearing more similarity to the ground truth reference. Interestingly, latent features such as clouds or shade thought to be lost to saturation re-manifest with noise.

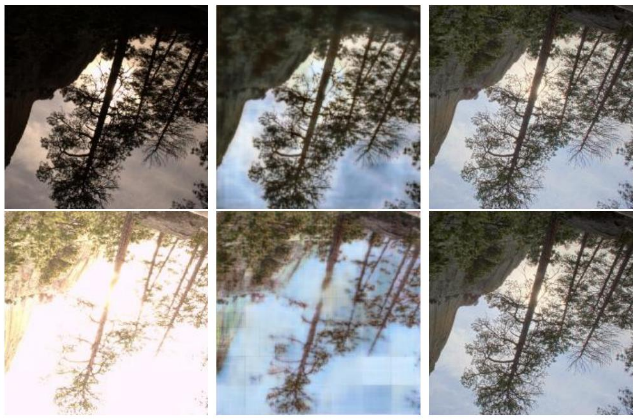

Fig. 8. Under and overexposed input reconstructions. Left to right: Input (x), Reconstruction ($\hat{y}$), and Ground Truth (y).

As shown in figure Figure 8, the model performs well on certain overexposed images, where it is able to recover the some of the lost details. It also recovered part of the tree trunk and leaves in the example shown above.

The Fairchild HDR dataset contained many outdoor scenes of nature, which exemplify a common problem in outdoor photography: overexposure due to sunlight. The consequence is a trade-off where the sky appears completely white, *or* the sky appears normal but other objects appear dark. Because these scenes were pervasive throughout the training dataset, the model tends to overgeneralize between disparate overexposed regions poorly, filling them with a light blue color as in Figure 9; in that example, the dome at the top of the building appears translucent to the sky with green foliage at its edges.

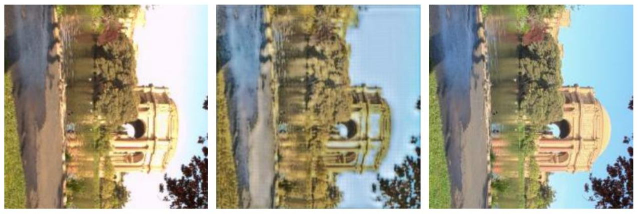

Fig. 9. Feature reconstruction error on dome. Notice that the model appears to act as a 'sky-filler', replacing even parts of structures with the most common true color of overexposed regions: sky-blue. Left to right: Input (x), Reconstruction ($\hat{y}$), and Ground Truth (y).

Another limitation of our model was its failure to effectively recover high resolution detail from images, often leading to hazy outlines or ghostly artifacts in regions with object boundaries. Checkerboard and block artifacts were more prevalent in earlier model iterations, but were reduced by lower the filter sizes of deconvolution (upsampling) layers and with the introduction of more trainable weights.

6 Conclusion

Our CNN autoencoder model functions as a proof-of-concept for the use of neural networks in the global reconstruction of HDR images with latent regional features from strictly LDR input images, which will only become a problem of greater importance as the quantity of photo sensors increases. Compared with existing algorithmic tone-mapping methods, our image quality was poorer, the colour and gradients being noisier, and features less sharp. The autoencoder achieved a maximum test dataset PSNR of 15.05 dB, lower than the 27.81 dB of the baseline render with OpenCV.

We believe that with a much larger dataset that encompasses expansive and diverse collections of real-life scenes, a neural network similar to ours would be able to surpass conventional inverse-tone mapped algorithms and achieve results comparable to multi-exposure tone mapped results. Their weights can carry great memory of image features from the scenes they were trained with, and can infer details that are suggested but not present in the inputs. This conclusion is supported by the measurable rise in validation metrics that we witnessed whenever more training examples were introduced.

Future improvements may come from the incorporation of hybrid network components, such as the use of GANs to achieve photorealistic output, discriminating for image quality metrics such as the PSNR metric itself, measurements of sharpness and natural colours. With greater computing capability, it would also be worthwhile to visualize, train and tune individual convolutional layers.

See our functioning implementation on GitHub at `https://github.com/BayBenj/cs230-proj`.

Contributions

All three group members collaborated on paper-writing, experimenting with and analyzing various model parameters. Additionally:

Vincent performed data pre-processing and data augmentation (converting raw format to jpeg, scaling, cropping, etc.), implemented model updates (additional encoder layers, additional skip connection, dropout layers), error analysis (catching image alignment issue), worked on hyperparameter tuning.

Rohit implemented skip-connections based network, implemented PSNR metric, loss functions (L1, L2, total variation), implemented initial image alignment function, experimented with leaky ReLU activation, bilinear upsampling. implemented and experimented with regularization and VGG-16 trainable/frozen layers

Benjamin implemented initial VGG-16 loading, led diagram creation for training, development, and test loss, PSNR, and SSIM, oversaw the model topology diagram, formalized all our loss equations, implemented batch normalization in model, used in each layer of the HDR decoder, decoupled main driver from model code, led poster design, located the majority of cross-references.

References

1. S. Mann, "Compositing multiple pictures of the same scene," in *Proc. IS&T Annual Meeting, 1993*, pp. 50–52, 1993.
2. D. Marnerides, T. Bashford-Rogers, J. Hatchett, and K. Debattista, "Expandnet: A deep convolutional neural network for high dynamic range expansion from low dynamic range content," in *Computer Graphics Forum*, vol. 37, pp. 37–49, Wiley Online Library, 2018.
3. G. Eilertsen, J. Kronander, G. Denes, R. Mantiuk, and J. Unger, "Hdr image reconstruction from a single exposure using deep cnns," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 6, 2017.
4. M. D. Fairchild, "The hdr photographic survey," in *Color and imaging conference*, vol. 2007, pp. 233–238, Society for Imaging Science and Technology, 2007.
5. K. M. Williams, R. W. Schulte, K. E. Schubert, and A. J. Wroe, "Evaluation of mathematical algorithms for automatic patient alignment in radiosurgery," *Technology in Cancer Research & Treatment*, vol. 14, no. 3, pp. 326–333, 2015.
6. G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.
7. P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103, ACM, 2008.
8. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
9. F. Chollet *et al.*, "Keras." https://keras.io, 2015.
10. Q. Huynh-Thu and M. Ghanbari, "Scope of validity of psnr in image/video quality assessment," *Electronics letters*, vol. 44, no. 13, pp. 800–801, 2008.
11. P. E. Debevec and J. Malik, "Recovering high dynamic range radiance maps from photographs," *ACM SIGGRAPH 2008 classes*, p. 31, 2008.
12. E. Reinhard and K. Devlin, "Dynamic range reduction inspired by photoreceptor physiology," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 11, no. 1, pp. 13–24, 2005.
13. Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.