# Transforming Grayscale landscape images to RGB through autoencoders

Carlos Iván Fonseca Mondragón | A01771689

**ABSTRACT** This report presents the development and implementation of two convolutional landscape colorization models: an initial autoencoder baseline optimized with pixel-level Mean Absolute Error (MAE) and an enhanced model trained with the Structural Similarity Index Measure (SSIM). Both models operate on grayscale-to-RGB reconstruction of landscape images and share a standardized preprocessing pipeline. The baseline autoencoder learns basic structural and illumination cues but fails to recover semantically meaningful colors due to reconstruction-loss constraints and latent-space bottlenecks. In contrast, the enhanced model introduces perceptual optimization and improved architectural capacity, yielding significantly sharper structures, natural color distributions, and higher perceptual quality, with test SSIM values ranging from 0.8595 to 0.9866. Together, these implementations illustrate the progression from low-level pixel reconstruction to perceptually grounded colorization, highlighting the importance of structural similarity objectives and architectural enhancements for realistic image generation.

## 1. INTRODUCTION

The field of computer vision has increasingly focused on techniques that can restore or enhance visual information. Among these techniques, **autoencoders** have proven to be some of the most powerful tools for reconstructing complex image patterns. In this context, colorizing grayscale images represents a challenging task, as it requires the model to deduct color distributions from limited information.

This work examines two stages of model development. The first stage implements a convolutional autoencoder trained solely through pixel-level MAE, compressing grayscale features into a low-dimensional latent space and reconstructing them in RGB. While this model achieved stable convergence and a validation MAE of 0.1367, it produced washed-out silhouettes and lacked semantic color reasoning.

To handle these issues, a second model was developed with improvements in its architectural depth, increased representational capabilities and an SSIM-based optimization to better capture perceptual structure. This enhanced approach yields reconstructions with finer details, more coherent color distributions and higher perceptual fidelity across test samples.

Both models share a preprocessing workflow that includes resizing and normalization to enable efficient training on landscape imagery. The comparison

between them demonstrates the impact of loss function choice and architectural design on colorization quality, providing a clear pathway for future improvements such as perceptual losses, multiscale feature extraction, and extended training schedules.

## 2. DESCRIBING THE DATASET

The "*Landscape Color and Grayscale Images*" dataset, uploaded by user Bhabuk Ghimire on Kaggle.com, contains a total of 14,258 images in .jpeg format, each with a resolution of 150×150 pixels, with a combined size of 203.47 MB.

The dataset is evenly divided into two folders: "color" and "gray", each containing 7,129 images. Every image pair shares the same numeric label (from 0 to 7,128), which uniquely identifies corresponding color and grayscale versions of the same landscape.

None of the elements provided in the dataset are corrupted or are of a different resolution than the specified above.

## 3. DATA EXTRACTION

### *3.1 Preparing the dataset*

A TensorFlow-based routine was developed to load and preprocess all images from the dataset. The two main folders from the original dataset were placed in a shared folder, resulting in two directories: (gray/) and (color/). Each image was read using the tf.io.read_file() method and decoded via tf.image.decode_jpeg(), automatically assigning one channel for grayscale inputs and three channels for color inputs.

## 4. DATA TRANSFORMATIONS

### *4.1 Image rescaling*

After loading, all images were rescaled to a uniform resolution of 64×64 pixels using tf.image.resize_with_pad(), ensuring dimension consistency across the dataset. The chosen resize method was nearest neighbor, which preserves local contrast and edge definition, rather than performing interpolation-based smoothing.

Both integer (uint8) and float32 tensor versions were generated for each image:
The integer version represents the raw pixel data (0–255).
The float version corresponds to the normalized representation (0-1), used for model training.

Each processed image was stored in new folders (gray_processed/ and color_processed/) to maintain a clean division between raw and transformed data.

This resulted in a total size reduction average of approximately 80% for both gray and color images, going from the initial 203.47 MB, down to 41.3 MB, which dramatically increased the model's training speed.

Finally, a subset of 28 image pairs was manually moved from the procs into new directories (gray_test/ and color_test/) to serve as unseen samples for manual evaluation of the model's performance.

### *4.2 Train, test, and validation splits*

After rescaling, the dataset was divided into training, validation, and test subsets using the train_test_split() function from scikit-learn. Although implemented in a separate script, this operation is conceptually part of the Load phase of the ETL process.

A fixed randomization seed of 42 was used to ensure reproducibility. First, the dataset was split into a temporary subset (80%) and a test subset (20%). Then, the temporary subset was further divided into training (64%) and validation (16%) sets, resulting in a final distribution of approximately **64%** for training, **16%** for validation, and **20%** for testing.

This division guarantees that the model is evaluated on unseen data, ensuring a fair and reliable analysis of its generalization performance.

## 5. MODEL IMPLEMENTATION

### *5.1 Input Data*

After preprocessing and splitting the data into training, validation, and testing, the model uses grayscale landscape images as inputs and their corresponding color versions as target outputs. Each image has a fixed resolution of 64×64 pixels. Unlike a regression or classification model, this implementation learns spatial and chromatic relationships from pixel intensities. The input tensors have a shape of (64, 64, 1), while output tensors have a shape of (64, 64, 3). Values of each pixel were normalized to facilitate training stability.

### *5.2 Model Architecture and Assumptions*

This autoencoder was implemented through the use of TensorFlow and Keras. As it stands, this architecture is capable of learning latent space representations from unlabeled image data. It consists of two main components:

- **Encoder**: The encoder is in charge of compressing the initial grayscale image into a low-dimensional representation.
    - The encoder uses Conv2D layers with 32, 64, 128 and 256 filters, each followed by a use of BatchNormalization() and a ReLU activation.
    - Downsampling is applied through convolutional strides of 2.
    - Obtained feature maps are flattened and passed through a dense layer of 128 neurons, defining the **latent dimension (LATENT_DIM = 128)**.
    - Finally, a Dropout layer with a 0.5 rate is applied to avoid overfitting.

- **Decoder**: The decoder reconstructs the color version of the image through the latent representation.
    - It starts with a dense layer, reshaped into (8, 8, 256), followed by multiple Conv2DTranspose layers, that gradually upsample the image back to the original size (64×64), while having 3 layers corresponding to the RGB spectrum, the result is a (64, 64, 3) image.

    - Each layer mirrors the encoded structure, using ReLU activations and BatchNormalization() for stability.
    - The final output employs a sigmoid activation to normalize color values.

The encoder and decoder were combined into a sequential model and compiled utilizing the Adam optimizer and Mean Absolute Error (MAE) as the loss function, given its utility for image processing tasks.

Training was set to 135 epochs, with a batch size of 128, using the following callbacks:

- ModelCheckpoint (saving the best model based on validation loss)
- ReduceLROnPlateau (reducing learning rate when progress stalls)
- EarlyStopping (stopping training after 10 patience epochs with no improvement).

*5.3 Model Objective and the MAE formula*

The model aims to learn the mapping between grayscale and color representations by reducing the pixel-wise reconstruction error. This can be represented through the Mean Absolute Error formula:

$$L = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i|$$

**Figure 1**: Mean Absolute Error formula

Where $\hat{y}_i$ represents the reconstructed color image output delivered by the decoder. Reducing this loss value makes the model reproduce chromatic tones and gradients, learning colorization patterns while being unsupervised.

## 6. MODEL EVALUATION

The performance of the autoencoder is evaluated primarily through the training and validation loss, both measured using Mean Absolute Error (MAE). MAE quantifies the average absolute difference between the predicted color pixel values and their ground-truth counterparts. Lower MAE values indicate more accurate color reconstruction, as the model produces color outputs closer to the original images.

In addition to the observed losses during training, this implementation also reports an MAE value after running a query.

## 7. MODEL RESULTS

Following model training and evaluation, the autoencoder demonstrated stable reconstruction performance, measured through Mean Absolute Error (MAE). Although training was set to 135 epochs, the EarlyStopping mechanism halted the process after epoch 37 after the validation loss stopped improving, ensuring that the model maintained generalization and did not drift into overfitting.

At the final epoch (Epoch 37), the results were:

- Training loss (MAE): 0.1291
- Validation loss (MAE): 0.1367
- Learning rate: $5 \times 10^{-4}$

For comparison, during the first epoch, the model began with:

- Training loss (MAE): 0.2010
- Validation loss (MAE): 0.2171
- Learning rate: $1\times10^{-3}$

This reduction in error across the first 37 epochs indicates that the autoencoder successfully optimized its reconstruction capabilities, learning a compressed latent representation capable of partially restoring color information.

The decrease from an initial MAE of around 0.20 to around 0.13 shows that the model improves upon its ability to approximate the ground truth image. The gap between the training (0.1291) and validation (0.1367) losses remains small, implying that the model is not suffering from overfitting. However, its capacity is limited, as the training error stops improving early and settles quickly.

Qualitative evaluation provides the best picture of performance in colorization models. The following figure compares:

- Input grayscale image
- Ground truth image
- Model output with its corresponding MAE.


**Figure 2:** Evaluation of three unseen images in various lighting conditions.

The current model successfully reconstructs basic shapes and scene layout, captures tonal regions (such as a “dark forest”) and produces color intensities slightly consistent with the illumination.

The current issues with this first model are:

- Fine textures and high-frequency detail are lost
- Colors often appear washed out, monochromatic, or incorrect.
- Buildings, trees, and structural details lack clear definition.
- Complex scenes yield higher MAE values.

In the examples shown, MAE values ranged from 0.08 to 0.18, but even low-MAE results lack color diversity. This, however, is expected because MAE measures pixel-level deviation, not actual realism.

Overall, the model produces silhouette-level approximations rather than actual color reconstructions. It mainly captures illumination (dark vs. light regions), not the actual color tones of the original scene.

A line plot was generated to visualize MAE across train and validation sets. Where validation values appear to show no progress past around epoch 20, training MAE keeps decreasing, albeit at a very slow pace.

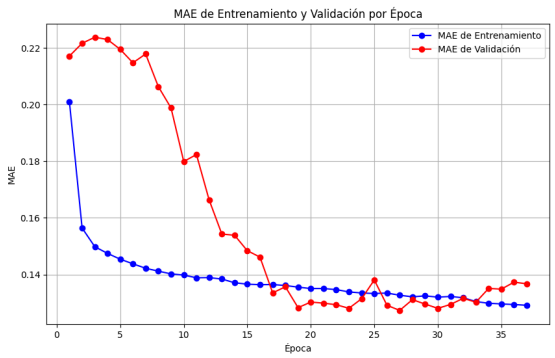


**Figure 3:** Train vs. Validation MAE line plot.

Plotting the loss curves confirms that early stopping was appropriate, as continued training would not reduce error further. This suggests that the model rapidly reaches the limits of what it can learn under its architectural constraints.

## 8. CHANGES TO THE MODEL

The model seen from this point onward replaces the previous encoder-encoder architecture for a U-Net based one, which includes skip connections via feature map concatenation. The architecture maintains four spatial levels 64×64, 32×32, 16×16, 8×8) and each encoder level connects directly to its decoder counterpart. The bottleneck is twice as large (256 dimensions vs. 128).

This architectural difference translates into the new model's ability to preserve spatial information that was otherwise lost previously.

### *8.1 Model architecture and improvements*

This autoencoder was implemented through the use of TensorFlow and Keras, incorporating several architectural improvements over the baseline model. It consists of two main components with lateral connections:

- **Encoder**:
  - The encoder uses Conv2D layers with 32, 64, 128, and 256 filters progressively.
  - The initial convolutional layer employs LeakyReLU activation with an alpha value of **0.1** to prevent dying ReLU during early model training, subsequent layers us a regular ReLU activation.
  - Each convolutional layer followed is followed by the use of BatchNormalization() for training stability.
  - Downsampling is applied through a stride value of **2** in the second through fourth layer, reducing spatial layers from 64×64 to 8×8.
  - Feature maps from layers **x1 through x3** are preserved for later concatenation with decoder layers, implementing a skip connection mechanism.
  - The resulting feature maps are flattened and passed through a dense layer of 256 neurons, defined by the variable **LATENT_DIM = 256**, which has twice the capacity of the baseline model.
  - Dropout layers have reduced rates (0.1-0.2), which help in preventing overfitting without

compromising the quality of reconstructed images.

- **Decoder:**
  - The decoder starts with a dense layer that shapes the latent vector to 8×8×256, to then be reshaped into a spatial dimension (8,8,256).
  - Multiple Conv2DTranspose layers upsample the image back to its original 64×64 dimension through three upsampling stages.
  - In each upsampling stage, the decoder concatenates the upsampled features with the corresponding encoder feature maps (**x3, x2, x1)** using the Concatenate() function, making use of the U-Net skip connections.
  - The concatenation results in an increase in input channels for subsequent layers, but also provides high-frequency information that was otherwise lost in the previous model.
  - Each upsampling layer employs ReLU activations and BatchNormalization().
  - The final convolutional layer generates three channels, with each one belonging to an element in the RGB spectrum, utilizing a sigmoid activation to normalize color values from 0 to 1.

The encoder and decoder were compiled into a pipeline using Keras's Functional API (rather than Sequential, used in the baseline model) to enable the skip connections. The model was compiled with the Adam optimizer and a custom **Structural Similarity Index (SSIM)** loss function, representing a significant shift from the baseline MAE approach.

Training was configured for 50 epochs with a batch size of 32, implementing the following callbacks:

- ModelCheckpoint (saving the best model based on validation loss)
- LearningRateScheduler (implementing a three-phase schedule: 1e-4 for epochs 0-29, 5e-5 for epochs 30-59, 1e-5 thereafter)
- EarlyStopping (stopping training after 15 patience epochs with no improvement).

*8.2 Model Objective and the SSIM Loss Function*

The model aims to learn the mapping between grayscale and color images by optimizing for structural similarity, rather than the pixel-wise reconstruction error seen with the baseline MAE (Mean Absolute Error).

The Structural Similarity Index (SSIM) measures perceptual similarity between two images by comparing luminance, contrast, and structure:

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

**Figure 4:** SSIM formula used in the model.

Where:

- $l(x, y)$ is the luminance comparison = $\frac{2\mu_x\mu_y + C_1}{\mu_x^2+\mu_y^2 + C_1}$
- $c(x, y)$ is the contrast comparison = $\frac{2\sigma_{xy} + C_2}{\sigma_x^2+\sigma_y^2 + C_2}$
- $s(x, y)$ is the structure (covariance) comparison = $\frac{\sigma_{xy} + C_3}{\sigma_x+\sigma_y + C_3}$

With typical values $\alpha = \beta = \gamma = 1$ and $C_1$, $C_2$, $C_3$ as small constants to avoid division by zero.

The model's loss function is defined as:
**1 - (1/N) Σ SSIM(y_true, y_pred)**

This loss function differs from MAE in several aspects:

- SSIM calculates similarity in local windows, capturing spatial correlation that is lost with MAE.
- SSIM considers luminance, contrast and structural information separately, where MAE treats all pixel errors equally.
- While MAE penalizes any sort of deviation linearly, SSIM is more tolerant to spatial changes as long as structural information is preserved.

By employing SSIM as the model's loss function, the model chooses to preserve structural details over pixel-wise accuracy. This is specially useful in the context of image colorization, where a single input can have multiple plausible color interpretations, and structural similarity is more important than exact numerical precision.

*8.3 Technical Foundations and Theoretical Improvements*

The enhanced model implements several design principles based on recent research:

**Skip Connections and Information Flow**: The use of skip connections addresses the bottleneck seen in the baseline model. When downsampling, high-frequency spatial information (such as edges, textures, and other fine details) are progressively lost. By concatenating encoder and decoder features directly at the same spatial resolutions, the model can avoid the bottleneck, thus preserving more detail in the final reconstruction, and, at the same time, mitigating vanishing gradient issues.

This design is inspired by U-Net (Ronneberger et al., 2015), which demonstrated superior performance in image-to-image translation tasks.

**Regularization Strategy:** Reduced dropout rates (0.1-0.2, compared to 0.3-0.5 in the baseline model) show a balance between overfitting prevention and maintaining reconstruction abilities, as excessive levels of dropout eliminate key information required to build an accurate prediction output. The absence of dropout in the decoder allows for all available information (from both the bottleneck and skip connections) to be used for colorization.

**Bottleneck Capacity:** The latent dimension doubling, from 128 to 256 reduces the compression ratio from approximately 96:1 to 48:1. This increase in capacity translates into the ability to represent more complex color patterns and distributions, which, in turn, allows for a

better representation of landscapes with diverse color palettes.

**Batch Size:** A reduced batch size was applied (32 vs. 128), which has been shown to improve generalization (Keskar et al., 2017). While this change increases training time per epoch, it results in a better final performance.

## 9. ENHANCED MODEL EVALUATION

### *9.1 Evaluation Methodology*

The performance of the enhanced autoencoder is evaluated through the SSIM (Structural Similarity Index), which measures perceptual similarity on a scale from -1 to 1, where 1 indicates perfect structural similarity. The SSIM loss used during training is defined as (1 - SSIM), thus lower values indicate better performance. SSIM values above 0.80 generally indicate good perceptual quality, while values above 0.90 suggest excellent reconstruction.

Model performance is tracked at four stages:

1. **Training monitoring**: SSIM loss is tracked on the training and validation sets after each epoch.
2. **Model selection**: The checkpoint with the lowest validation loss is selected.
3. **Test evaluation:** The final model is assessed with a separate test data split.
4. **Query evaluation**: When running a query through the graphical interface provided in the repository, an SSIM score is shown below the reconstructed image.

### *9.2 Training Dynamics*

The model is trained with two distinct phases:

- In the initial phase, from epochs 0-30, there is rapid SSIM improvement as the model learns basic color patterns. Skip connections allow for a faster convergence than the baseline model with a 1e-4 learning rate.
- In the intermediate to final phase, the model goes through a gradual refinement and fine-tuning phase with a reduced 5e-5 learning rate.

### *9.3 Comparative Analysis*

Although SSIM is a better performance indicator for image colorization than the previous MAE metric seen in the baseline, a qualitative assessment of the results is still the best way to evaluate real world results. Beyond numerical metrics, the following details are examined:

- Color realism: Natural, plausible colors for landscapes.
- Detail preservation: Fine textures and edges maintained via skip connections.
- Color consistency: Spatial coherence without color bleeding.
- Saturation: Appropriate vibrancy (MAE models tend toward desaturation).

## 10. ENHANCED MODEL RESULTS

### *10.1 Training performance*

Following model training, the enhanced autoencoder demonstrated increased reconstruction performance as measured

through SSIM Loss. The model was configured for 50 epochs with early stopping capability (patience=15), though it completed all planned epochs as validation loss continued to improve steadily throughout training.

**Final Training Results (Epoch 50):**

- Training loss (SSIM): 0.0329
- Validation loss (SSIM): 0.0424
- Learning rate: $5\times10^{-5}$
- Best validation checkpoint: Epoch 49 (validation loss: 0.0422)

**Initial Performance (Epoch 1):**

- Training loss (SSIM): 0.3784
- Validation loss (SSIM): 0.7140
- Learning rate: $1\times10^{-4}$

The dramatic reduction from the initial SSIM loss of approximately 0.38-0.71 to 0.033-0.042 demonstrates that the enhanced model successfully learned to reconstruct images with high structural fidelity. After converting the loss values to SSIM index scores, the final model achieves SSIM ≈ 0.967 on training data and SSIM ≈ 0.958 on validation data, indicating excellent performance across dataset.

The training process exhibited three distinct phases:

- Epochs 1-14: In early training, validation loss quickly dropped from 0.7140 to 0.0454 (93.6% reduction).
- Epochs 15-30: Steady refinement, validation loss decreased to 0.0426.
- Epochs 31-50: Fine-tuning at reduced learning rate ($5\times10^{-5}$), achieving minimum validation loss of 0.0422 at epoch 49.

The small gap between train and validation sets (0.009) confirms solid generalization without overfitting.

*10.2 Loss Curve Analysis*

The SSIM loss curves show a smooth convergence with minimal variance variability. Unlike the baseline, where validation MAE showed no progress past epoch 20, the enhanced model's validation loss continued improving until epoch 49.

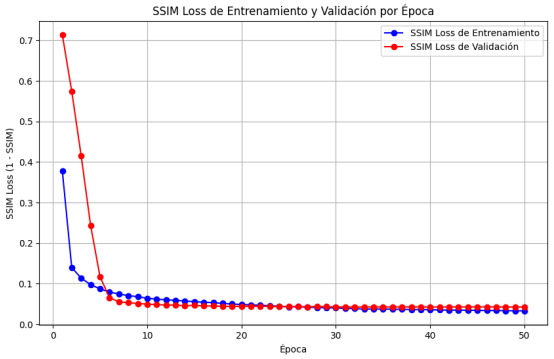


**Figure 5:** Train vs. Validation SSIM line plot.

*10.3 Qualitative Results and Improvements*

Qualitative evaluation reveals considerable improvements over the baseline. The following figures compare:

- Input grayscale image
- Ground truth image
- Model output with its corresponding SSIM.

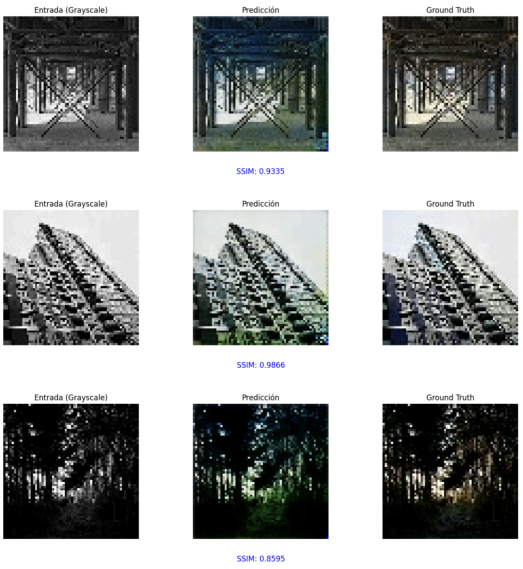


**Figure 6:** Grayscale, Prediction and Ground Truth images plot.

A visual inspection of the new model's predictions shows clear improvement over the baseline across all scene types and lighting conditions, with more coherent spatial layouts, more naturally balanced colors, minimal bleeding and a strong preservation of fine details.

*10.4 SSIM Score Interpretation and Performance Analysis*

SSIM values achieved across test samples fall within the 0.86-0.99 range, with an average of 0.9265.

Variations in SSIM correlate with scene difficulty (such as limited chromatic cues) and not with model deficiencies. When testing scenes with simpler geometry and clearer illumination, results show a higher level of visual similarity, while low-light regions reduce SSIM, even if outputs are not subjectively considered bad.

MAE from the baseline is not comparable: it rewards pixel-level accuracy but fails to capture perceptual realism, which explains why outputs from the baseline looked monochromatic and flat. SSIM better reflects human visual judgment and aligns with the improved visual quality of the enhanced model.

Test results closely match the validation SSIM (0.958), showing good generalization without overfitting. Remaining limitations arise from the inherent ambiguities of grayscale colorization and the 64×64 resolution constraints, not from architectural flaws.

## 11. FUTURE ENHANCEMENTS

Several changes could strengthen the current model's performance. First, extending the number of epochs in training would allow the network to benefit from the unused final section of the learning rate scheduler, a longer training period with adaptive decay could refine convergence and improve perceptual accuracy, especially in complex scenes. Second, increasing the model's capacity, either through more encoder-decoder stages or with multiscale feature extraction could allow a better semantic reasoning and reduce color ambiguities, Third, increasing the resolution of input and output images to, for example, 128×128 pixels will allow for preservation of finer detail, at the cost of a much higher level of processing time and power needed. Finally, incorporating perceptual or hybrid loss functions may further improve texture fidelity and stabilize color consistency beyond what SSIM alone provides.

## 12. CONCLUSION

This study evaluated an enhanced autoencoder architecture for grayscale-to-color image reconstruction, achieving stable training and strong perceptual similarity, with a validation SSIM of 0.958 and real-life test scores between 0.8595 and 0.9866. The model preserves spatial structure, produces coherent color distributions and maintains finer details that were previously lost.

In contrast to the baseline, which was limited by MAE optimization and highly prone to blurred, monochrome silhouettes, the new model delivers a more natural and accurate colorization. The use of SSIM as a loss function when training enables better structural alignment and reduces color bleeding, while the expanded latent capacity improves detail preservation and mitigates the earlier model's inability to infer meaningful colors.

Despite these gains, there are still challenge to approach. Low-light scenes lack a sufficient level of chromatic detail, and the 64×64 resolution constrains the amount of detail that can be captured by the autoencoder. Nevertheless, the improved design establishes a superior baseline for image colorization. Future work could integrate a higher amount of epochs, multiscale skip connections or perceptual-adversarial losses to further enhance semantic reasoning and robustness across complex scenes.

## 13. REFERENCES


a. Johnson, J., Alahi, A., & Fei-Fei, L. (2016). *Perceptual losses for real-time style transfer and super-resolution*. In European Conference on Computer Vision (ECCV).
b. https://arxiv.org/abs/1603.08155
c. Lyashenko, V., & Jha, A. (2025, April 25). *Cross-validation in machine learning: How to do it right.* Neptune AI. https://neptune.ai/blog/cross-validation-in-machine-learning-how-to-do-it-right
d. Zhang, R., Isola, P., & Efros, A. A. (2016). *Colorful Image Colorization*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
e. Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). *Image quality assessment: From error visibility to structural similarity*. IEEE Transactions on Image Processing, 13(4), 600–612.
f. Wang, Z., & Bovik, A. C. (2009). *Mean squared error: Love it or leave it? A new look at signal fidelity measures*. IEEE Signal Processing Magazine, 26(1), 98–117.
g. Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-Net: Convolutional networks for biomedical image segmentation*. arXiv preprint arXiv:1505.04597.