INDIAN INSTITUTE OF TECHNOLOGY, GUWAHATI

Mehta Family School of Data Science and Artificial Intelligence

Project report on

Image Colorization

Group **#20**

Submitted to:
Dr. Debanga Raj Neog

Submitted by:
Sri Harsha R(224101048)
Subramanian V(224101050)
Neha Dhuttargaon(224101037)
Binayak Behera(224101014)
Pratik Rana(224101040)

For the course fulfillment of **DA526**: Image Processing with Machine Learning

ACKNOWLEDGEMENT

This project is being submitted as a requirement for course fulfillment of DA526 - Image Processing with Machine Learning.It is a pleasure to acknowledge our sense of gratitude to Dr. Debanga Raj Neog who guided us throughout the project work. His timely guidance and suggestions were encouraging. We thank the Teaching Assistants who were always helpful in clearing doubts. Finally, we thank our classmates for their support.

Sri Harsha R(224101048)
Subramanian V(224101050)
Neha Dhuttargaon(224101037)
Binayak Behera(224101014)
Pratik Rana(224101040)

Contents

1. Abstract **1**
2. Problem Statement **1**
3. Motivation **1**
4. Contribution **2**
5. Literature Survey **2**
6. Methodology **4**
6.1. Introduction 4
6.2. Color Space 5
6.3. From B&W to color 6
6.4. Architecture 7
Table 1. Encoder Network 7
Table 2. Fusion Network 8
Table 3. Decoder Network 8
6.5. Preprocessing 9
6.6. Encoder 9
6.7. Feature Extractor 9
6.8. Fusion 10
6.9. Decoder 10
6.10. Objective Function and Training 10
7. Why Inception Resnet V2? **11**
8. Dataset **12**
9. Experiments and Results **12**
10. Accuracy **13**
11. Conclusion and Future Work **13**
12. References **14**

List of Figures

1. Welsh - Transfers color from a related reference image……………………………..…. 2

2. Levin - User-guided color using scribbles…………………………………………..……. 3

3. End-to-end Convolutional Neural Network colorization………………………………….. 3

4. Zhang, R.(2017) - end-to-end CNN approach incorporating user "hints" in the spirit of scribble based methods allowing real-time use…………………………………………… 4

5. Grayscale Image Representation………………………………………………………… 4

6. RGB Color Channels……………………………………………………………………… 5

7. RGB color Encoding……………………………………………………………………… 5

8. f() is the neural network, [B&W] is our input, and [R],[G],[B] is our output……………… 5

9. Lab Color Space…………………………………………………………………………... 6

10. B&W to color……………………………………………………………………………... 6

11. An overview of the model architecture…………………………………………………. 7

12. Stacking the luminance component three times …………………………………….. 9

13. Fusing the Inception embedding with the output of the convolutional layers of the encoder………………………………………………………………………………… 10

14. Inception-ResNet-v2 Results………………………………………………………….. 11

15. Output…………………………………………………………………………………… 12

16. Training loss……………………………………………………………………………. 13

1. Abstract

Our project focuses on colorizing gray-scale images using state-of-the-art deep learning techniques. Taking inspiration from recent advancements in this field, we have implemented a model that combines a deep Convolutional Neural Network (CNN) trained from scratch with high-level features extracted from the pre-trained Inception-ResNet-v2 model. By utilizing a fully convolutional architecture, our encoder-decoder model is capable of effectively processing images of varying sizes and aspect ratios.

Furthermore, we have created a diverse range of applications for our model, with a particular emphasis on colorizing historical photographs. This allows us to bring vibrant and lifelike colors to black-and-white images from the past, preserving their historical significance while also providing a visually captivating experience.

2. Problem Statement

Given a Black-and-White image color it using a deep Convolutional neural Network and Inception-Resnet-v2 pre-trained network.

3. Motivation

The process of coloring gray-scale images can have a significant impact in various domains, such as historical image restoration and surveillance footage enhancement. Gray-scale images contain limited information, but adding color components can provide more insights into the image's semantics and help convey its meaning better. In fields such as medicine, colorization can help doctors and researchers better analyze images such as X-rays and MRIs. By adding color to these images, they can better differentiate between different types of tissue and identify anomalies.

Deep learning models such as Inception, ResNet, and VGG are typically trained using colored image datasets. However, when these networks are applied to gray-scale images, the results may not be as accurate or effective. To overcome this limitation, a prior colorization step can be performed to improve the results.However, designing and implementing an effective and reliable system that automates this process still remains a challenging task. The difficulty increases even more if the goal is to fool the human eye.

One reason why automating the colorization process is difficult is because it requires a deep understanding of color perception and how humans interpret visual information. Moreover, different colors can have different meanings in different contexts, which makes it challenging to create a system that can accurately colorize images across different domains.
Despite the challenges, there has been significant progress in the field of image colorization in recent years, particularly with the use of deep learning techniques. Researchers have developed various

algorithms that can automatically colorize gray-scale images, with some achieving impressive results that are difficult to distinguish from naturally colored images.

We have implemented a model that can colorize images to a certain degree by combining a deep Convolutional Neural Network architecture with the latest Inception model, Inception-ResNet-v2. This model is based on Inception v3 and Microsoft's ResNet and serves as a high-level feature extractor, providing information about the image contents that aids in colorization.Although we trained the deep CNN from scratch, the size of our training dataset was small due to time constraints, limiting our model's ability to colorize a wide range of images. Nonetheless, our results demonstrate that it is possible to automate the colorization process, and we have validated some approaches used by other researchers.

4. Contribution

In brief, our contribution in this report can be summarized as follows.

- High-level feature extraction using a pre-trained model (Inception-ResNetv2) to enhance the coloring process.
- Analysis and intuition behind a colorization architecture based on CNNs.
- Test of the model with historical pictures.
- Code documentation is available in Github repo.

5. Literature Survey

In 2002, Welsh et. al. presented a novel approach which was able to colorize an input image by transferring the color from a related reference image.

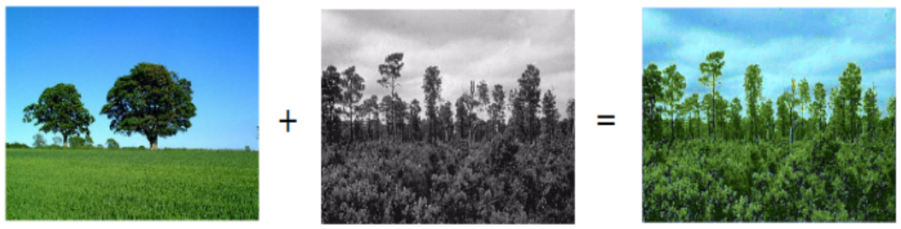

Fig 1. Welsh - Transfers color from a related reference image

Subsequent improvements of this method were proposed, exploiting low-level features and introducing multi-modality on the pixel color values .

In parallel, another research line was initiated in 2004 by Levin et. al., who proposed a scribble based method which required the user to specify the colors of a few image regions.

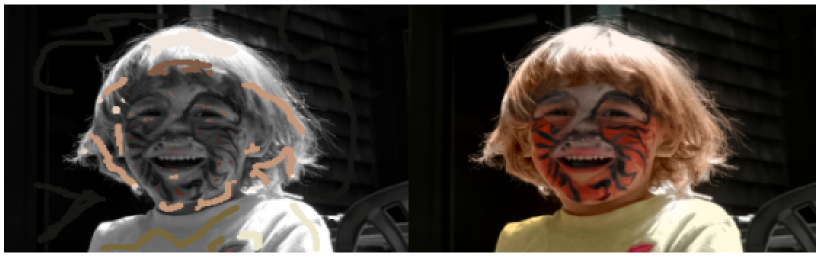

Fig 2 . Levin - User-guided color using scribbles

This colorization methodology aroused the interest of animators and cartoon-aimed techniques were proposed [10, 11]. The results from these approaches were certainly impressive at that time, however, the results were highly dependent on the artistic skills of the user. More recently, automated approaches have been proposed. For instance, in [12] Desphande et al. conceived the coloring problem as a linear system problem.

In recent years, CNNs have been shown through experimental studies to nearly halve the error rate for object recognition [13], leading to a significant shift towards deep learning within the computer vision community. To this end, Cheng Z. and colleagues [14] introduced a deep neural network that utilized image descriptors (such as luminance, DAISY features [15], and semantic features) as inputs. In 2016, Iizuka, Serra, and others [16] presented an approach that used global-level and mid-level features to encode and colorize images, which served as inspiration for our model's architecture and validation. However, we also integrate a pre-trained model into our approach. Recent similar approaches have also been proposed, such as the multi-modal scheme developed by Zhang and colleagues [17], where each pixel was assigned a probability value for every possible color.

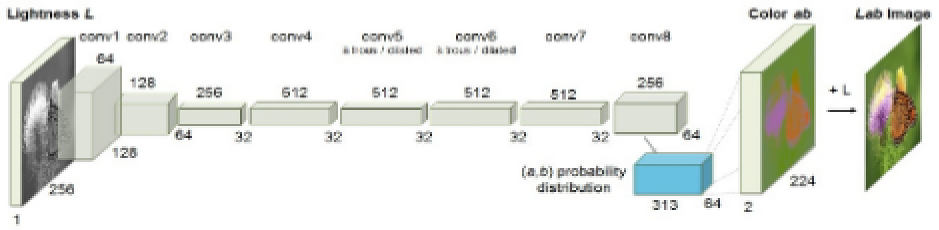

Fig 3. End-to-end Convolutional Neural Network colorization

Additionally, Larsson and colleagues [18] created an interesting approach that used a fully convolutional version of VGG-16 [19], with the classification layer removed, to construct a color probability distribution for each pixel.

A recent study by Zhang and colleagues introduced an end-to-end CNN approach that incorporates user "hints," similar to scribble-based methods. This system also includes a color recommender system designed to aid novice users. The authors claim that their approach enables real-time usage of their colorization system. This research highlights the continued advancement in this field of study.

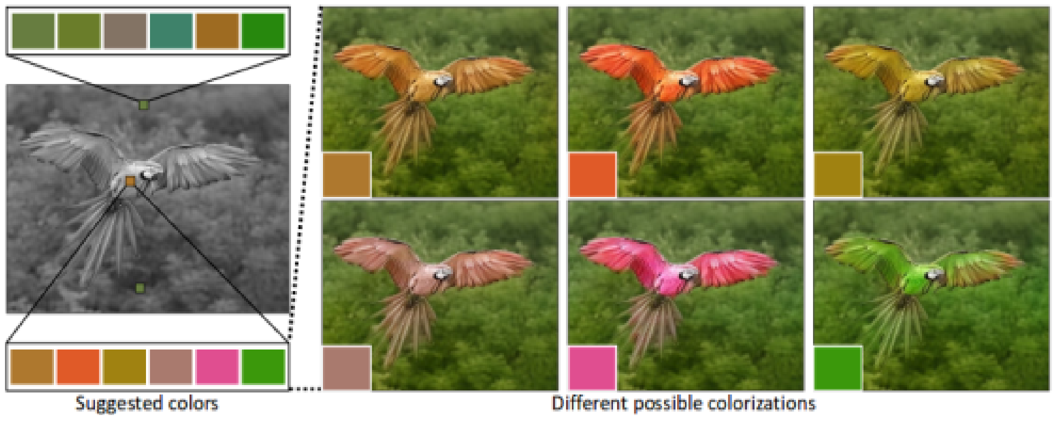

Fig 4. Zhang, R.(2017) - end-to-end CNN approach incorporating user "hints" in the spirit of scribble based methods allowing real-time use

6. Methodology

6.1. Introduction

Black and white images can be represented in grids of pixels. Each pixel has a value that corresponds to its brightness. The values span from 0–255, from black to white.

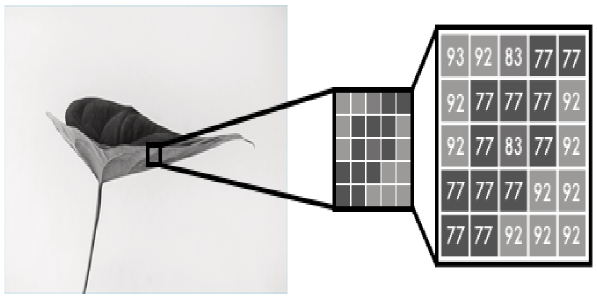

Fig 5. Grayscale Image Representation

Color images consist of three layers: a red layer, a green layer, and a blue layer. This might be counter-intuitive to you. Imagine splitting a green leaf on a white background into the three channels. Intuitively, you might think that the plant is only present in the green layer.But, as you see below, the leaf is present in all three channels. The layers not only determine color, but also brightness.

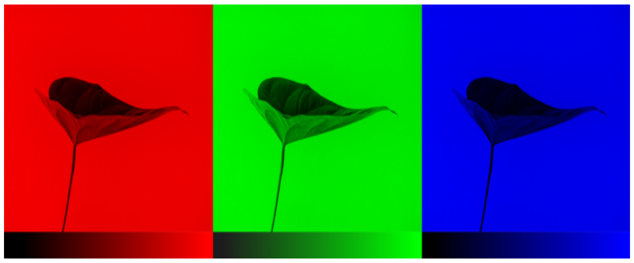

Fig 6. RGB Color Channels

To achieve the color white, for example, you need an equal distribution of all colors. By adding an equal amount of red and blue, it makes the green brighter. Thus, a color image encodes the color and the contrast using three layers:

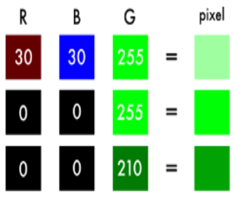

Fig 7. RGB color Encoding

Just like black and white images, each layer in a color image has a value from 0–255. The value 0 means that it has no color in this layer. If the value is 0 for all color channels, then the image pixel is black.
A neural network creates a relationship between an input value and output value. To be more precise with our colorization task, the network needs to find the traits that link grayscale images with colored ones. In sum, we are searching for the features that link a grid of grayscale values to the three color grids.

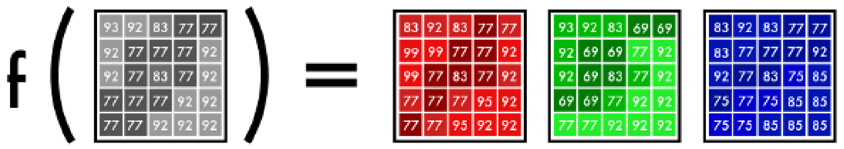

Fig 8. f() is the neural network, [B&W] is our input, and [R],[G],[B] is our output.

6.2. Color Space

we'll use an algorithm to change the color channels, from RGB to Lab. ***L*** stands for lightness, and ***a*** and ***b*** for the color spectra green–red and blue–yellow.

As you can see below, a Lab encoded image has one layer for grayscale, and has packed three color layers into two. This means that we can use the original grayscale image in our final prediction. Also, we only have two channels to predict.

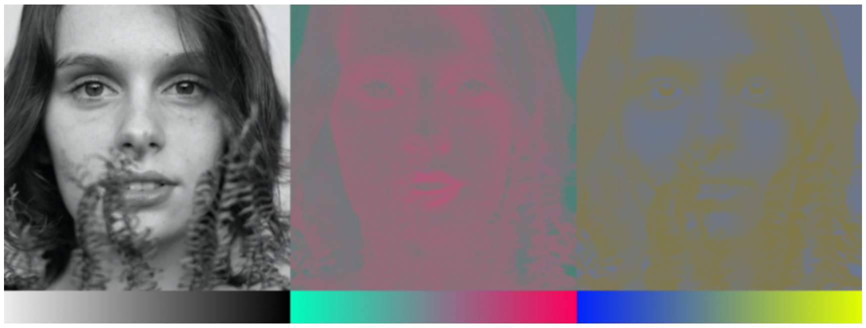

Fig 9. Lab Color Space

6.3. From B&W to color

Our final prediction looks like this. We have a grayscale layer for input, and we want to predict two color layers, the **ab** in **Lab**. To create the final color image we'll include the **L**/grayscale image we used for the input. The result will be creating a **Lab** image.

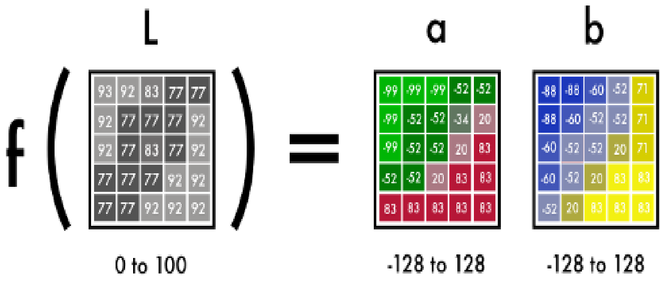

Fig 10. B&W to color

We consider images of size ***H* × *W*** in the **CIE L*a*b*** color space. Starting from the luminance component $X_L \in$ R H×W×1 , the purpose of our model is to estimate the remaining components to generate a fully colored version $X \in R^{H\times W\times 3}$. In short, we assume that there is a mapping F such that

$$\mathbf{F : X_L \longrightarrow (X_a, X_b),}$$

where X_a, X_b are the a*, b* components of the reconstructed image, which combined with the input give the estimated colored image $X = (X_L, X_a, X_b)$.
In order to be independent of the input size, our architecture is fully based on CNNs, a model that has been extensively studied and employed in the literature. In brief, a convolutional layer is a set of small learnable filters that fit specific local patterns in the input image. Layers close to the input look for simple patterns such as contours, while the ones closer to the output extract more complex features.

As already pointed out, we choose the CIE L*a*b* color space to represent the input images, since it separates the color characteristics from the luminance that contains the main image features. Combining the luminance with the predicted color components ensures a high level of detail on the final reconstructed image.

6.4. Architecture

Given the luminance component of an image, the model estimates its a*b* components and combines them with the input to obtain the final estimate of the colored image. Instead of training a feature extraction branch from scratch, we make use of an Inception-ResNetv2 network (referred to as Inception hereafter) and retrieve an embedding of the gray-scale image from its last layer. The network architecture we implement is illustrated in Fig. 1 with slight modifications with the dimensions.

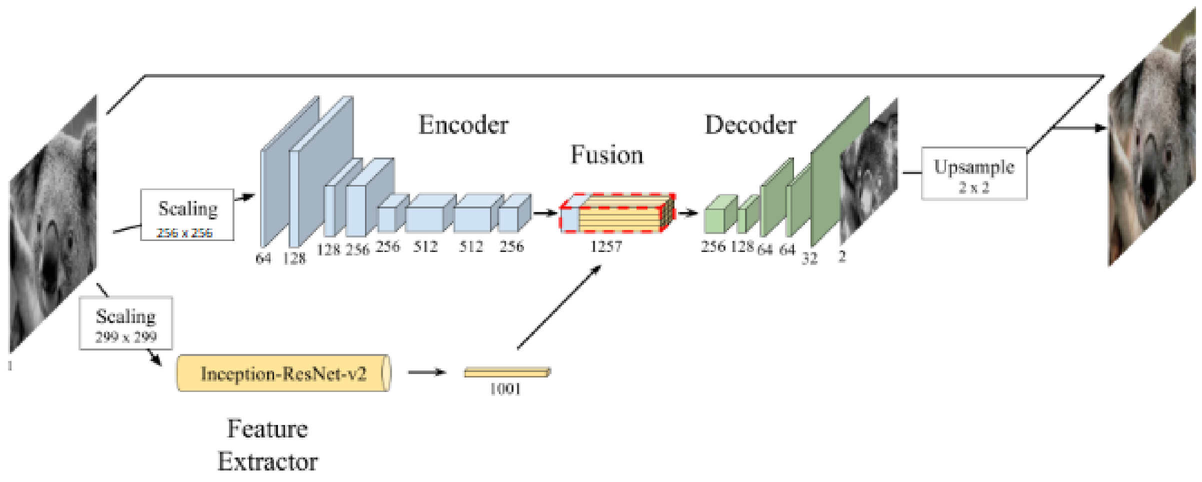

Fig 11. An overview of the model architecture.

The network is logically divided into four main components. The encoding and the feature extraction components obtain mid and high-level features, respectively, which are then merged in the fusion layer. Finally, the decoder uses these features to estimate the output.The following tables further details the network layers.

Table 1. Encoder Network

Layer	Kernels	Stride
conv	128 × (3 × 3)	1
max pooling	2 x 2	2
conv	128 × (4 × 4)	1
conv	128 × (3 × 3)	1
max pooling	2 x 2	2

conv	256 × (4 × 4)	1
conv	256 × (3 × 3)	1
max pooling	2 x 2	2
conv	256 × (4 × 4)	1
conv	256 × (3 × 3)	1
conv	256 × (3 × 3)	1

Table 2. Fusion Network

Layer	**Kernels**	**Stride**
conv	256 × (1 × 1)	1 x 1

Table 3. Decoder Network

Layer	**Kernels**	**Stride**
conv	128 × (3 × 3)	1
conv	64 × (3 × 3)	1
upsampling	2 x 2	1
conv	128 × (3 × 3)	1
upsampling	2 x 2	1
conv	64 x (4 x 4)	1
conv	64 x (3 x 3)	1
conv	32 x (2 x 2)	1

conv	2 x (3 x 3)	1
upsampling	2 x 2	1

Each Convolutional layer uses a ReLu activation function, except for the last one that employs a hyperbolic tangent function. The feature extraction branch has the same architecture of Inception-Resnet-v2, excluding the last softmax layer

6.5. Preprocessing

Image transformer: There is an instance of the `ImageDataGenerator` class, which is used to perform data augmentation on the input images. Data augmentation techniques such as shear, zoom, rotation, and horizontal flip are applied to the images during training. This helps increase the diversity and robustness of the training data.

In order to facilitate proper learning, the pixel values of all three image components are normalized by centering and scaling them according to their respective ranges. This results in values within the range of [-1, 1].

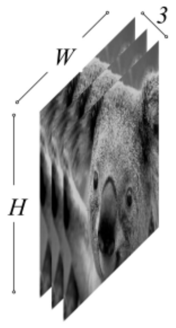

Fig. 12: Stacking the luminance component three times

6.6. Encoder

The Encoder is responsible for handling gray-scale images with dimensions of H x W, and produces a feature representation of dimensions H/8 x W/8 x 512. It achieves this through the utilization of eight convolutional layers, each equipped with a 3x3 kernel. Padding is employed to maintain the input size of each layer. Additionally, the first, third, and fifth layers use a stride of 2, which results in a halving of their output dimensions and consequently reduces the computational load required.

6.7. Feature Extractor

High-level features such as identifying whether an image is of an underwater scene or an indoor scene can provide useful information for the colorization process. To generate an image embedding, we utilized a pre-trained Inception model. Firstly, we resize the input image to 299x299 dimensions. Then, we duplicated the image to create a three-channel image, as depicted in Figure 2, which fulfills the dimension requirements of the Inception model. Subsequently, we input the resulting image to the network and retrieve the output of the last layer before the softmax function. This generated a 1001x1x1 embedding.

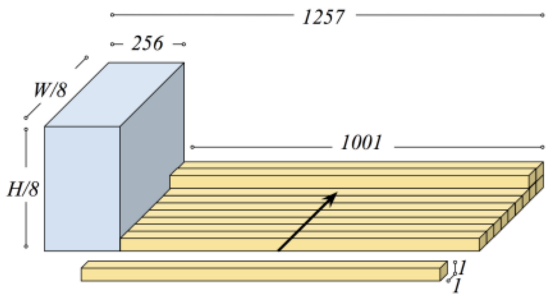

Fig. 13: Fusing the Inception embedding with the output of the convolutional layers of the encoder

6.8. Fusion

The fusion layer operates by taking the feature vector obtained from Inception and duplicating it 2 times HW/8. It then attaches this to the feature volume produced by the Encoder along the depth axis, as depicted in Figure 3. This yields a single volume that contains both the encoded image and the mid-level features, with dimensions of H/8 x H/8 x 1257. By replicating and concatenating the feature vector multiple times, we ensure that the semantic information it conveys is evenly distributed across all spatial regions of the image. This approach is also adaptable to arbitrary input image sizes, thereby enhancing the flexibility of the model. Lastly, we apply 256 convolutional kernels with dimensions of 1x1, resulting in a feature volume with dimensions of H/8 x W/8 x 256.

6.9. Decoder

The Decoder receives the feature volume with dimensions of H/8 x W/8 x 256 and utilizes a sequence of convolutional and up-sampling layers to generate a final layer with dimensions of H x W x 2. Up-sampling is accomplished using a simple nearest-neighbor approach, which doubles the height and width of the output relative to the input.

6.10. Objective Function and Training

The optimal model parameters are found by minimizing an objective function defined over the estimated output and the target output. In order to quantify the model loss, we employ the Mean Square Error between the estimated pixel colors in a*b* space and their real value. For a picture X, the MSE is given by,

$$C(\mathbf{X}, \boldsymbol{\theta}) = \frac{1}{2HW} \sum_{k \in \{a,b\}} \sum_{i=1}^{H} \sum_{j=1}^{W} (X_{k_{i,j}} - \tilde{X}_{k_{i,j}})^2,$$

where θ represents all model parameters, $X_{ki,j}$ and $X^1{}_{ki,j}$ denote the ij:th pixel value of the k:th component of the target and reconstructed image, respectively. This can easily be extended to a batch B by averaging the cost among all images in the batch, i.e. $1/|\mathcal{B}| \sum_{\mathbf{X} \in \mathcal{B}} C(\mathbf{X}, \boldsymbol{\theta})$
While training, this loss is back propagated to update the model parameters θ using Adam Optimizer with an initial learning rate η = 0.001. During training, we impose a fixed input image size to allow for batch processing.

- 90% training - 10% validation
- Test Batch size of 20 images
- ~23 hours training the network
- Keras with TensorFlow backend
- Stochastic Gradient Descent with Adam Optimizer(η= 0.001)

7. Why Inception Resnet V2?

Inception-ResNet-v2, a convolutional neural network (CNN) that achieves a new state of the art in terms of accuracy on the ILSVRC image classification benchmark. Inception-ResNet-v2 is a variation of the earlier Inception V3 model which borrows some ideas from Microsoft's ResNet papers.

As an example, while both Inception V3 and Inception-ResNet-v2 models excel at identifying individual dog breeds, the new model does noticeably better. For instance, whereas the old model mistakenly reported Alaskan Malamute for the picture on the right, the new Inception-ResNet-v2 model correctly identifies the dog breeds in both images.

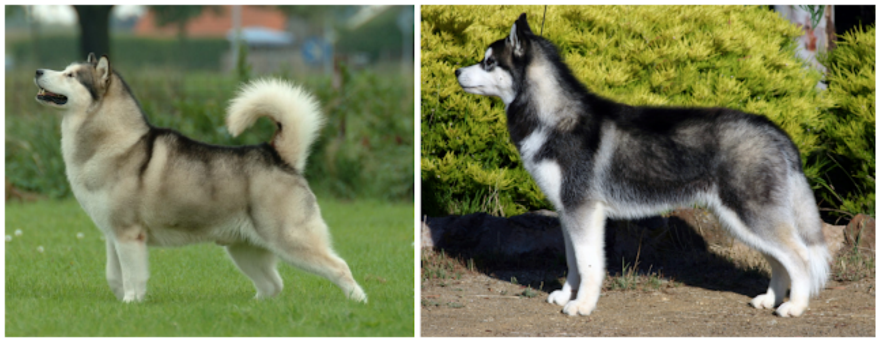

Fig 14. Inception-ResNet-v2 Results

8. Dataset

- The dataset includes around **2,000** classic painting images, including works by famous artists such as Monet and Van Gogh.
- The images are provided in color and will be decolorized to create training data for the neural network.
- The dataset has a size of **626.85 MB**, and there are **2,300** files in total.
- The images have various sizes, ranging from **256x256** to **5000 x 5000** pixels.
- The dataset also includes metadata about each painting, such as the artist, title, and year of creation.
- This dataset has been used in various computer vision tasks, such as image colorization, style transfer, and image classification.
- The images in this dataset are considered high-quality and are suitable for training deep learning models.

9. Experiments and Results

As important as the network's architecture itself is the choice of the dataset. The dataset is composed of many pictures of old paintings. We have used 90% of the dataset for training and 10% for testing, therefore all images in the training set are rescaled to 224 × 224 for the encoding branch input and to 299 × 299 for Inception. Each image gets stretched or shrunk as needed, but its aspect ratio is preserved by adding a white padding if needed.

- Near-photorealistic colorization for some images.
- Alternative but realistic colored estimates.
- Reduced dataset: the network performances depend on which global features are present in the test images

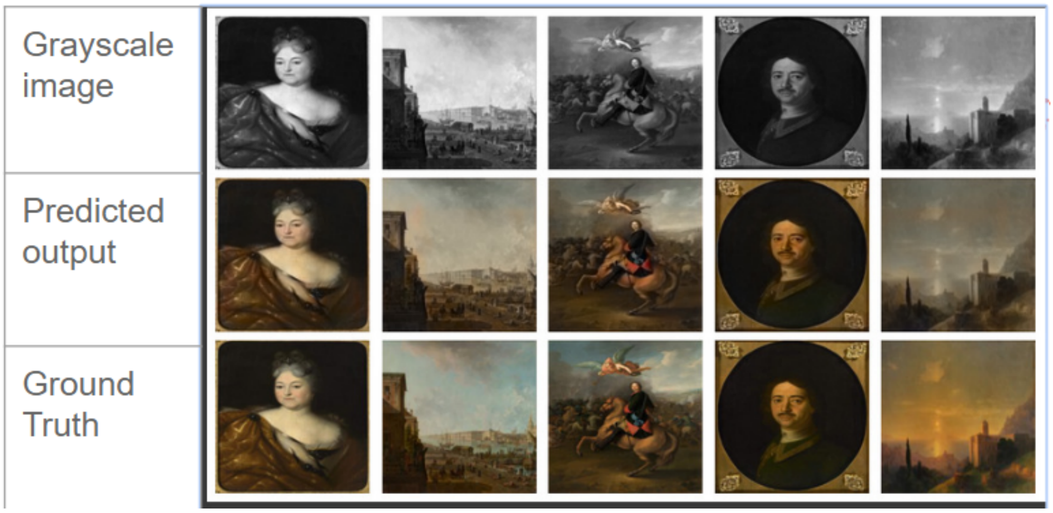

Fig 15. Output

10. Accuracy

Image colorization accuracy cannot be precisely quantified using traditional metrics due to the subjective nature of color perception and the lack of a definitive ground truth. The process involves artistic choices, cultural contexts, and multiple valid colorizations. Metrics like SSIM, PSNR, and color difference metrics provide insights but cannot capture subjective aspects. Evaluation relies on qualitative assessment, visual inspection, and human judgment, considering factors such as quality, naturalness, coherence, and subjective preference.

In our research, we have determined that it is not possible to calculate the accuracy of an image colorization model using traditional metrics. This is due to the subjective nature of color perception and the absence of a definitive ground truth for colorization. However, we still provide a means to gauge the model's accuracy by analyzing the training loss graph.

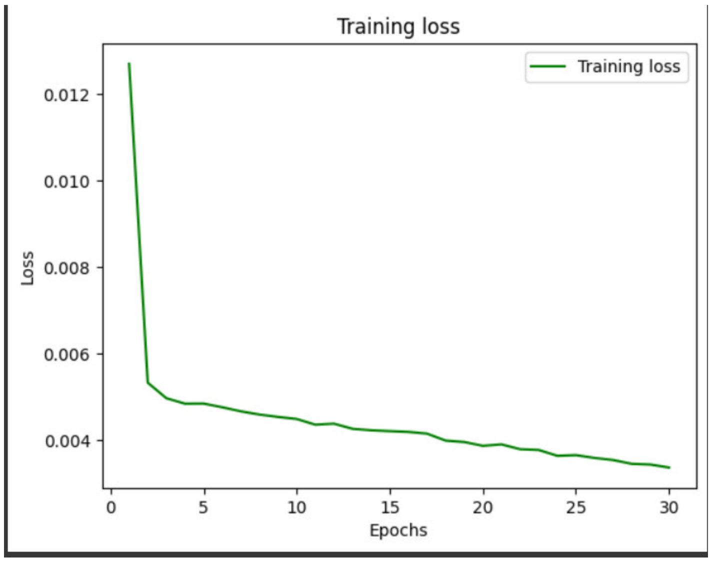

Fig 16. Training Loss Graph

11. Conclusion and Future Work

The above project showcases the potential of using deep learning architectures for image colorization tasks. Here are some further details on the project and possible future directions:

- **Successful coloring of high-level image components**: The proposed approach was able to accurately colorize high-level image components such as the sky, sea, and forests. This suggests that end-to-end deep learning architectures can be suitable for certain image colorization tasks. Need for improvement in small detail coloring: While the model was successful in coloring high-level image components, there is still room for improvement in coloring small details. This is an area where more research is needed to refine the deep learning architecture and improve the accuracy of colorization.
- **Limited dataset size**: The dataset used in this project was limited to a reduced subset of images,

specifically painting images. This limited dataset size may lead to decreased performance when dealing with unseen images with different contents. Training the model over a larger dataset could potentially improve its performance.

- **Probabilistic approach for better mapping**: A probabilistic approach similar to variational autoencoders could help improve the mapping between luminance and ab components. This approach could also allow for image generation by sampling from a probability distribution.
- **Adapting the network architecture for video sequences**: Expanding the colorization technique to video sequences could potentially be a future direction. This would require adapting the network architecture to accommodate temporal coherence between subsequent frames.
- **Potential for reducing supervised work**: While some degree of human intervention may be required for image colorization, it still has the potential to reduce hours of supervised work. This is particularly relevant for areas such as historical preservation and restoration, where old photographs and videos can be restored to their original color.

In conclusion, the project demonstrates the potential of deep learning architectures for image colorization tasks. While there is still room for improvement, future research could potentially refine the deep learning architecture and expand its application to video sequences. The ability to automate the colorization process has the potential to reduce hours of supervised work, making it a valuable tool in fields such as historical preservation and restoration.

Github Link - https://github.com/binayakbehera999/grp-20-image-colorisation

12. References

1. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. CoRR abs/1512.00567 (2015)
2. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR abs/1512.03385 (2015)
3. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014)
4. Szegedy, C., Ioffe, S., Vanhoucke, V.: Inception-v4, inception-resnet and the impact of residual connections on learning. CoRR abs/1602.07261 (2016)
5. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. CoRR abs/1603.05027 (2016)
6. Welsh, T., Ashikhmin, M., Mueller, K.: Transferring color to greyscale images. In: ACM Transactions on Graphics (TOG). Volume 21., ACM (2002) 277–280
7. Ironi, R., Cohen-Or, D., Lischinski, D.: Colorization by example. In: Rendering Techniques, Citeseer (2005) 201–210
8. Charpiat, G., Hofmann, M., Sch¨olkopf, B.: Automatic image colorization via multimodal predictions. Computer Vision–ECCV 2008 (2008) 126–139
9. Levin, A., Lischinski, D., Weiss, Y.: Colorization using optimization. In: ACM Transactions on Graphics (ToG). Volume 23., ACM (2004) 689–694
10. Qu, Y., Wong, T.T., Heng, P.A.: Manga colorization. In: ACM Transactions on Graphics (TOG). Volume 25., ACM (2006) 1214–1220
11. S`ykora, D., Dingliana, J., Collins, S.: Lazybrush: Flexible painting tool for handdrawn cartoons. In: Computer Graphics Forum. Volume 28., Wiley Online Library (2009) 599–608
12. Deshpande, A., Rock, J., Forsyth, D.: Learning large-scale automatic image colorization. In:

Proceedings of the IEEE International Conference on Computer Vision. (2015) 567–575
13. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. (2012) 1097–1105
14. Cheng, Z., Yang, Q., Sheng, B.: Deep colorization. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 415–423
15. Tola, E., Lepetit, V., Fua, P.: A fast local descriptor for dense matching. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE (2008) 1–8
16. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Let there be color!: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. ACM Transactions on Graphics (TOG) 35(4) (2016) 110
17. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: European Conference on Computer Vision, Springer (2016) 649–666
18. Larsson, G., Maire, M., Shakhnarovich, G.: Learning representations for automatic colorization. In: European Conference on Computer Vision, Springer (2016) 577– 593
19. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. ICLR (2015)
20. Zhang, R., Zhu, J.Y., Isola, P., Geng, X., Lin, A.S., Yu, T., Efros, A.A.: Realtime user-guided image colorization with learned deep priors. arXiv preprint arXiv:1705.02999 (2017)
21. Robertson, A.R.: The cie 1976 color-difference formulae. Color Research & Application 2(1) (1977) 7–11
22. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016) http://www.deeplearningbook.org.
23. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European conference on computer vision, Springer (2014) 818–833
24. Bora, D.J., Kumar Gupta, A., Ahmad Fayaz, K.: Unsupervised diverse colorization via generative adversarial networks. International Journal of Emerging Technology and Advanced Engineering (2015)
25. Nixon, M.S., Aguado, A.S.: Feature Extraction & Image Processing for Computer Vision. Elsevier Ltd (2002)
26. Hoffman, G.: Cielab colorspace. Technical report, University of Applied Sciences, Emden (Germany), http://docs-hoffmann.de/cielab03022003.pdf (2003)
27. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.A.: Striving for simplicity: The all convolutional net. CoRR abs/1412.6806 (2014)
28. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR abs/1412.6980 (2014)
29. Chollet, F., et al.: Keras. https://github.com/fchollet/keras (2015)
30. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Man´e, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Vi´egas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015) Software available from tensorflow.org.
31. Nickolls, J., Buck, I., Garland, M., Skadron, K.: Scalable parallel programming with cuda. Queue 6(2) (March 2008) 40–53