APS360

Artificial Intelligence Fundamentals

Colourizing Grayscale Images
Final Report

Sandy Dai - 1003277973
Abhishek Paul - 1002395188
Yang Xiu - 1002654950

Word Count: 2462

Introduction

Photographs play an important role in connecting humans to the past, telling stories, and sharing rare sightings. Color plays an important role in providing information, contrast, and detail. With over 100 years of photography that remained exclusively in grayscale, much of the details in historical images is lost [1]. Colourization by hand is often a long and tedious task that takes up to a month to complete on a single image [2]. The team aims to solve this problem by developing a model that can realistically colorize images, to bring new appreciation to grayscale photos through color. The goal of the model is to produce a plausible output of what an image in color might look like.

In order to colorize the images, a generative model is needed to learn features and be able to produce a whole new image with color. The team proposed a Generative Adversarial Network (GAN) trained with flower images to simplify the complexity of the images while allowing for a broad range of colours.

Illustration and Figure

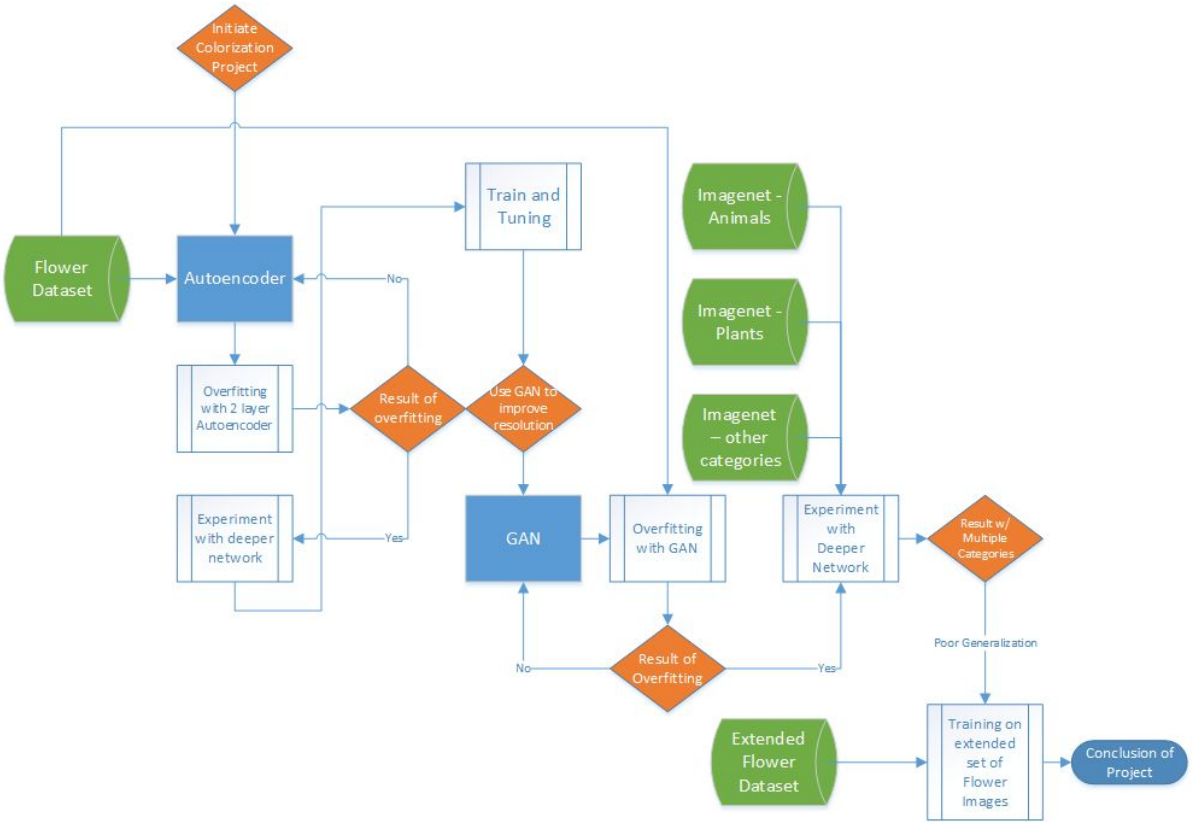

Initiate Colorization Project
Flower Dataset
Autoencoder
Train and Tuning
Imagenet - Animals
Imagenet - Plants
Imagenet – other categories
No
Overfitting with 2 layer Autoencoder
Result of overfitting
Use GAN to improve resolution
Yes
Experiment with deeper network
GAN
Overfitting with GAN
Experiment with Deeper Network
Result w/ Multiple Categories
Poor Generalization
No
Result of Overfitting
Yes
Extended Flower Dataset
Training on extended set of Flower Images
Conclusion of Project

Fig. 1 Process Flow Diagram of the project

Background and Related Work:

The use of machine learning for colorizing images is a well researched subject with many studies accomplishing it with variations of Autoencoders and GANs. One prominent work on the topic is *Colorful Colorization by Zang et al* [3]. They use an autoencoder but in order to avoid the destauration commonly associated with autoencoders, they create a custom loss function which emphasizes rare colors.

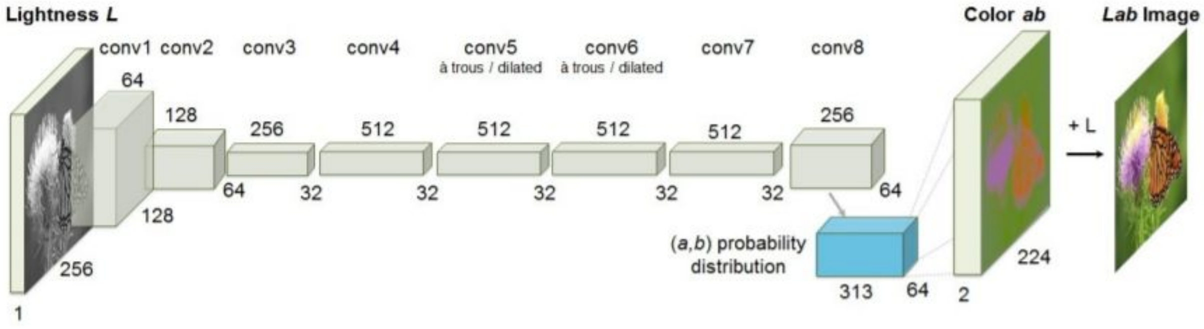


Fig. 2. Autoencoder Architecture, *Colorful Colorization [3]*

The model was trained on a large dataset from Imagenet and is able to generalize to many categories as seen below

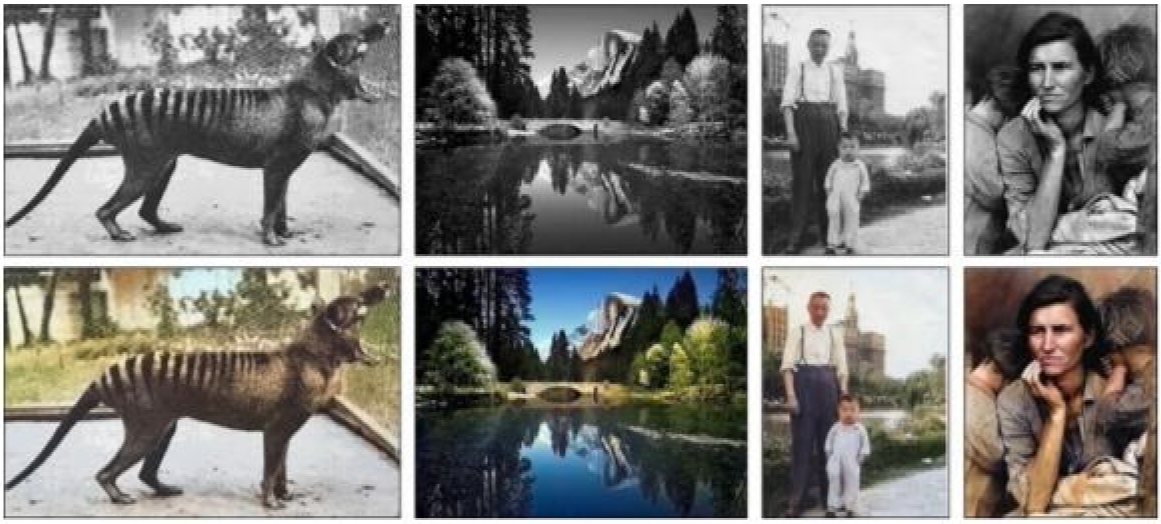


Fig. 3. Legacy Photo Reproductions, *Colorful Colorization* [3]

Another prominent work on colorizing is *Image Colorization using Generative Adversarial Networks [4]* by Nazeri et al. In this study they use a Conditional GAN with a UNET Autoencoder with skip connections.

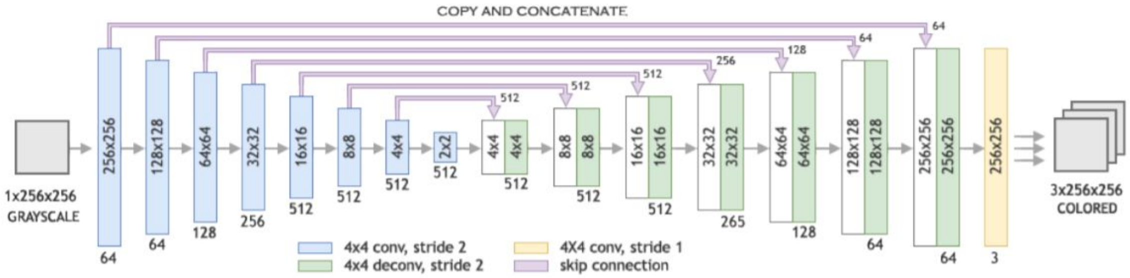


Fig. 4. UNET Model [4]

By using GAN with skip connections, the model is able to guarantee high resolution with the grayscale reproductions being almost identical to the original images.

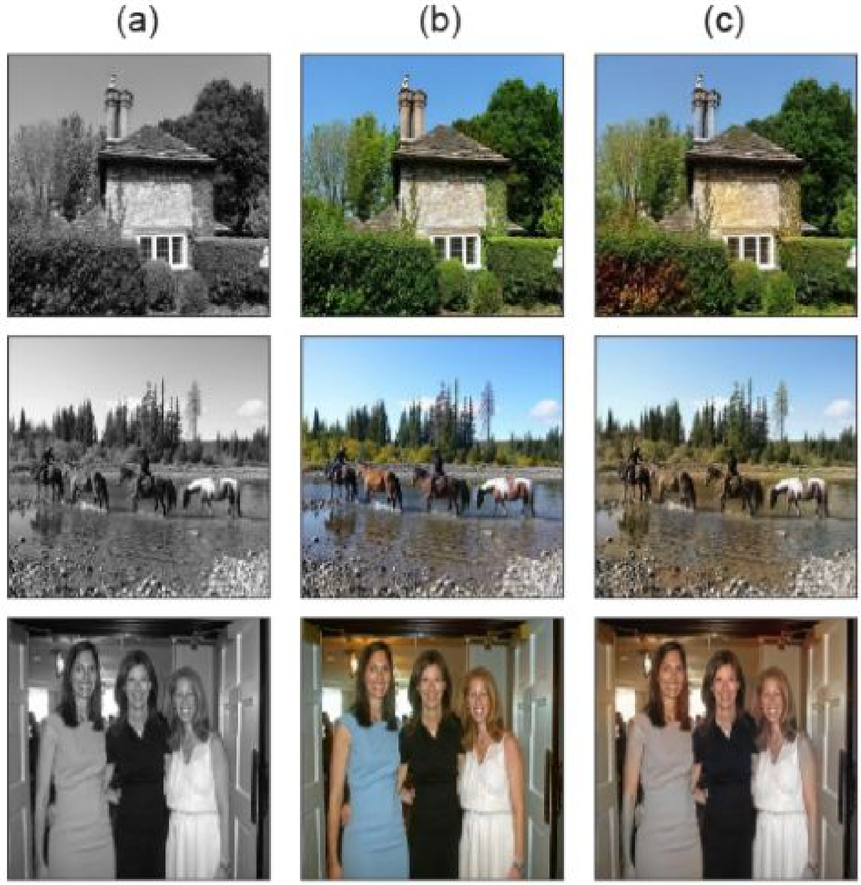


Fig. 5: For the three pictures:
(a) shows the gray scale input
(b) the original RGB image
(c) The reconstructed color image [4]

Data Processing:

We chose the Flower Dataset from the University of Oxford as it contained 8,189 flowers images of 102 different kinds of flowers [5]. For our validation set we used a 210 flower dataset from Kaggle [6], separate from the Oxford dataset.

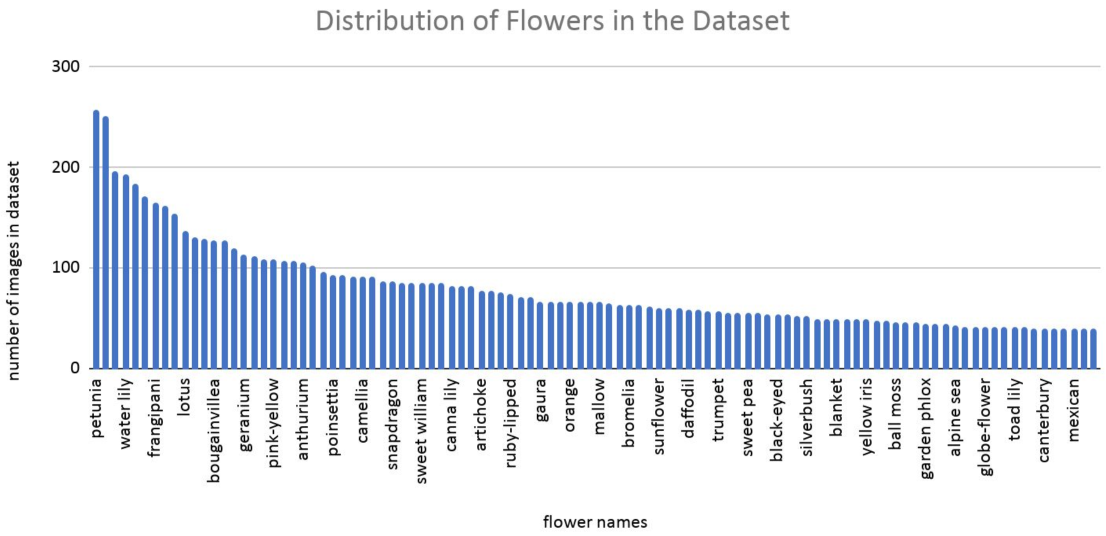
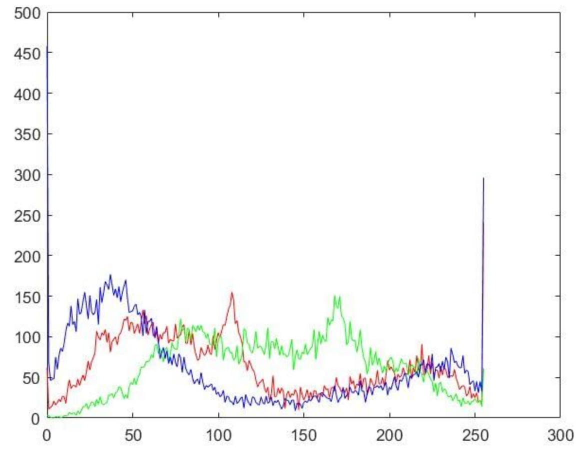
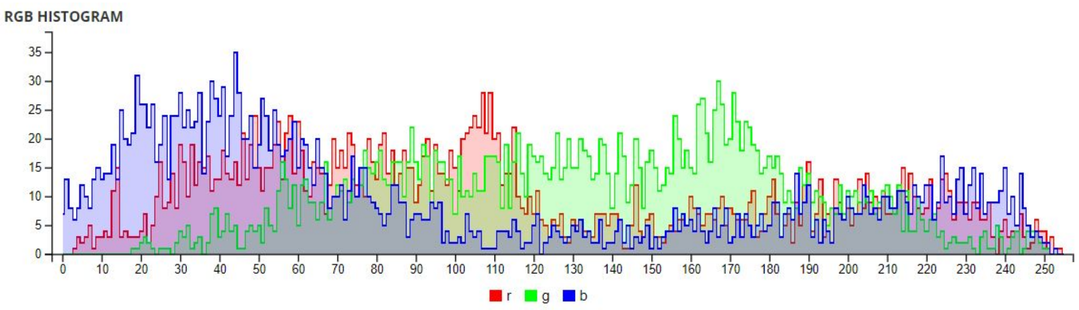


Fig. 6: Shows the distribution the different flowers in the dataset

The overwhelming amount of petunia and water lilies in the dataset leads to bias in purple colors but as seen color histograms below, the color distribution is otherwise balanced.



Figs. 7 and 8: Show the normalized color distribution histogram generated for our data set in Matlab



For data preprocessing we created a script which takes RGB images, resized them to specified dimensions and then converts them to grayscale. This allowed us to create our grayscale and RGB (input,target) pairs seen below:

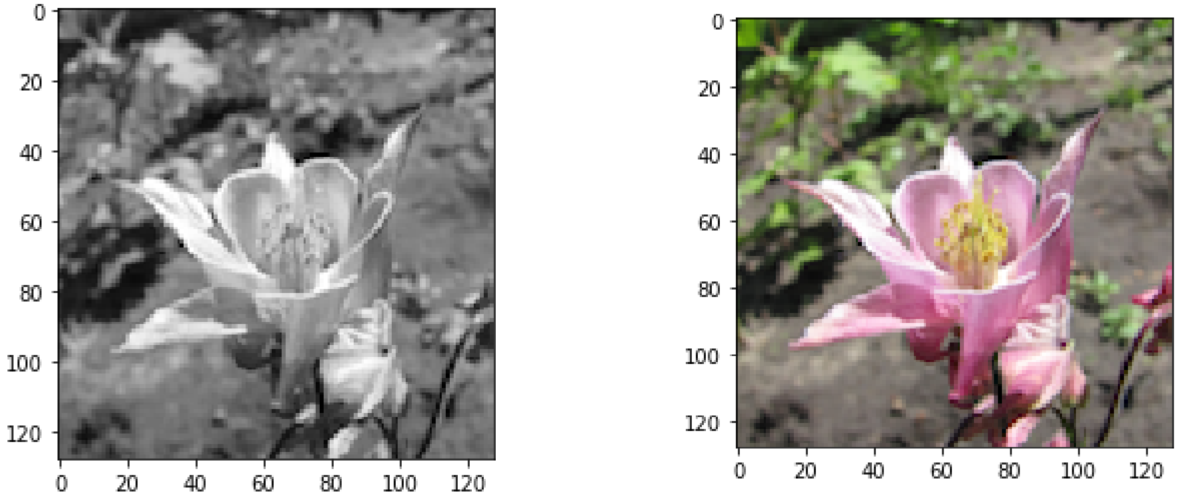


Fig 9. Shows the grayscale input along with its original RGB counterpart

The advantage of our data processing pipeline is that it allows us to easily switch between different dataset as it converts any image into the desired specifications.

Architecture

The final architecture is a GAN with a convolutional neural network (CNN) as discriminator and autoencoder as the generator. Downsampling is achieved using a stride of 2 in the encoder's convolutional layers. Poolings are not used due to their impact on the stability of GAN[7]. Each encoder layer except the first doubles the number of feature maps and batch normalizes the output. An embedding dimension of (16, 16, 512) is produced. Upsampling is achieved using a combination of skip connections and transposed convolutional layers. The ResNet style skip connection in conjunction with ReLU activation are implemented.

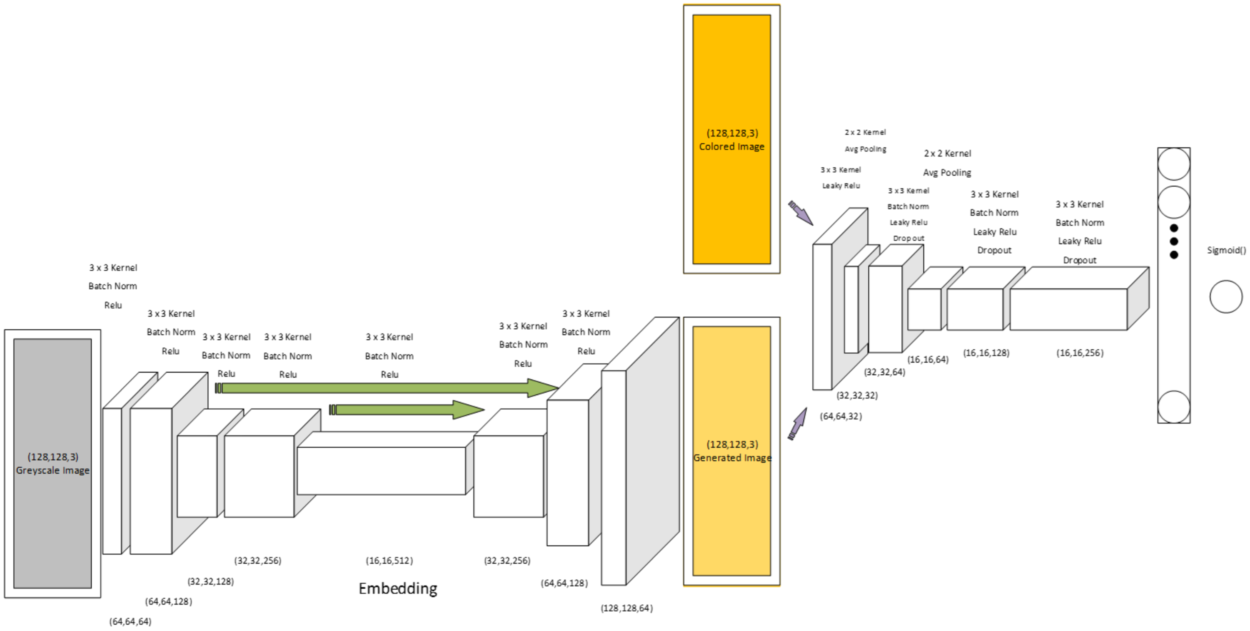


Fig 10. Architecture of GAN

The discriminator downsamples with (2 x 2) average pooling layers with a stride of 2. The outputs are also batch normalized. The discriminator uses LeakReLU and dropout layers. Embedding of dimension (16,16,512) is flattened and passed into a fully connected layer with one output.

Due to the fragile stability of GAN, regularization methods in both the architecture and the training function are non-trivial. In the training function, a threshold accuracy for the discriminator is set such that the discriminator is not trained when the accuracy is above the threshold. Noises are fed to the discriminator every 5th epoch to improve its generalizing capacity. The loss of the generator is modified to be a combination of binary cross entropy loss from the flipped labels and L1 loss from the original colored images.

Baseline Model

The baseline was chosen to be an autoencoder with convolutional layers in the encoder and 2 transposed convolutional layers in the decoder. Both downsampling and upsampling are achieved using strides and the embedding has dimensions (32,32,64). ReLU activations are implemented after each convolution and the sigmoid function is used to constrain output values. Standard (3x3) kernels were used in both the encoder and decoder.

Quantitative Result

There is no single objective loss function that can be used to evaluate the performance. There are three parameters that can be used to gain insight into the performance, namely, loss of discriminator and generator as well as the accuracy of the discriminator.

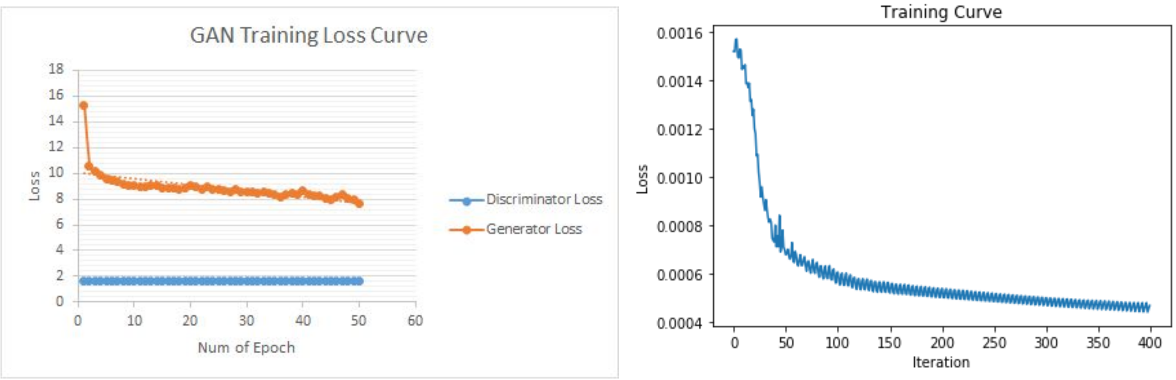


Fig 11. Training Curve of GAN (left) and autoencoder (right)

The training curve of both the discriminator and generator resembles the training curve of an autoencoder. Unlike regression and classification problems, a comparison between training loss and validation loss does not immediately prove overfitting. The loss may decrease as a result of displaying correct color for the features or displaying an average intensity of colors for all features. The latter leads to mode collapse. The exact value of the resultant training is also arbitrary because the loss function defined for the generator is a combination of binary cross entropy loss and L1 loss. The L1 loss is amplified by a factor of 75 in the training function. The loss value fluctuates with the choice of amplifying factor but the trend remains identical.

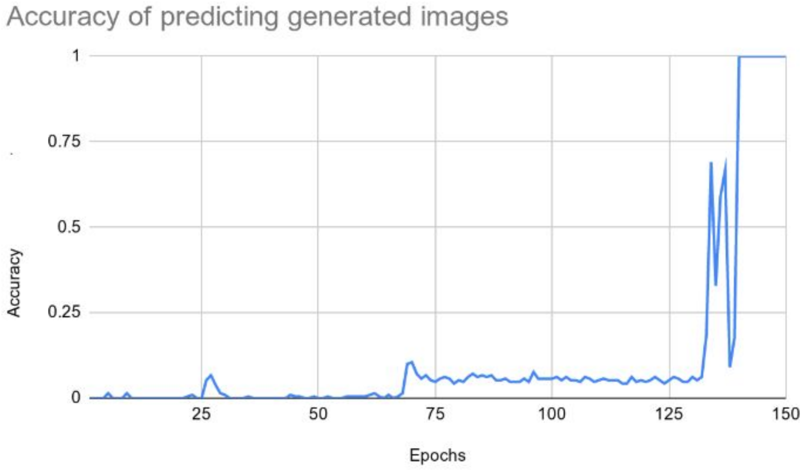


Fig 12. Prediction Accuracy of Discrimintor.

An appropriate quantitative measurement is the accuracy of the discriminator. The accuracy of the discriminator is evaluated by randomly sampling 100 images at the end of each epoch and record the number of images that the discriminator predicted correctly. Ideally, the accuracy should be evaluated over the entire dataset, but this is limited by the available computational power. Ideal tuning of the hyperparameters should indefinitely extend the region of oscillation so that neither does the discriminator or generator dominates one another. However, our model is eventually dominated by the discriminator despite the regularization methods implemented. This suggests further tuning is recommended. Therefore, given the prior of having a more powerful discriminator, the training is terminated shortly after the accuracy converges to 1. Further training will likely, but not always, lead to mode collapse.

Qualitative Result

Due to the lack of objective quantitative evaluation methods and the nature of generative models, qualitative results play an important role in determining performance. Training images were printed at fixed intervals.

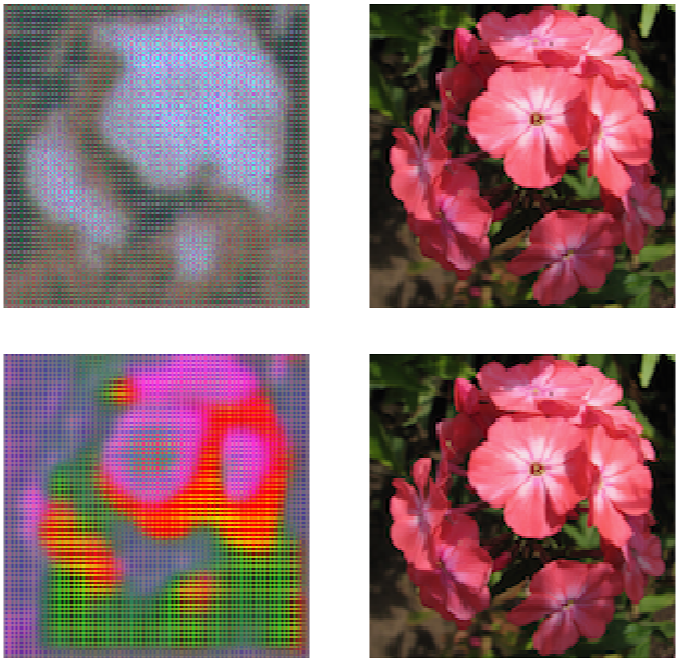


Fig 13. Results of underfitting on training data

Underfitting was relatively easy to spot. In early epochs, the pictures underwent drastic changes but rapidly converged. Overfittings are harder to detect. As mentioned in the quantitative results section, despite the declining training loss, mode collapse often occurred. Depending on the combination of other hyperparameters such as learning rate, momentum, discriminator threshold accuracy, etc. the generator loss decreased continuously after reaching perfect discriminator accuracy but the images generated remain visually identical.

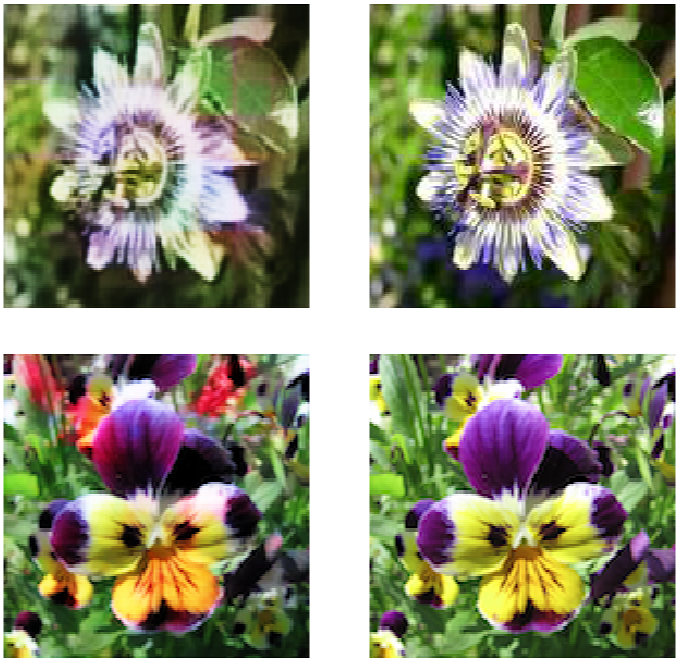


Fig. 14 Results of final model on training data

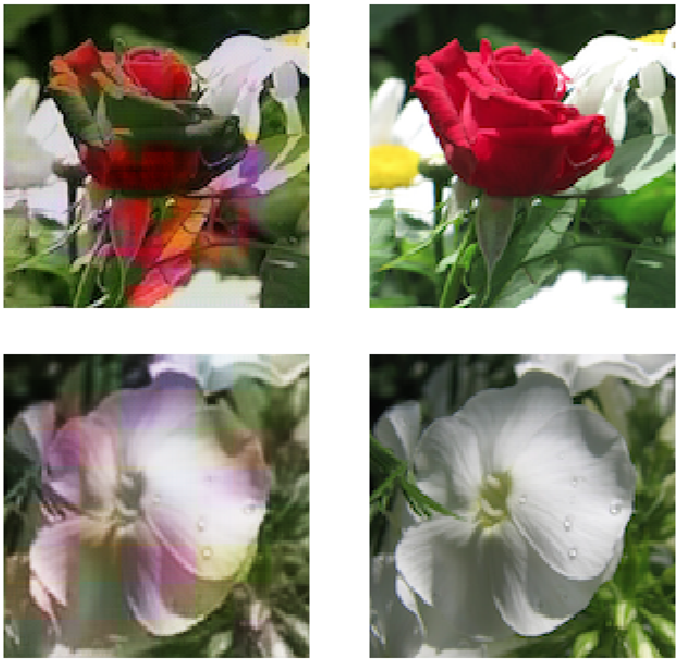


Fig. 15 Results of final model on validation data

In the absence of mode collapse, it becomes difficult to deduce a stopping point for training. Based on the result of validation data, the model naturally performs better on certain types than others and number of training epochs are determined by the sign of shifting in color in the validation data due to the bias in the dataset, e.g., when the white flower grows increasingly purple.

Evaluate Model on New Data:

Since our inputs are unlabeled images, evaluating new data is relatively simple. We created a function that can load images from links and then perform the appropriate data processing steps before running the images through our model.

As we mentioned before, the goal of the project is not a perfect reproduction of the RGB images from grayscale but instead a plausible reproduction. The model was first evaluated on some random images on the validation set and the results are seen below.

Testing Data Set:

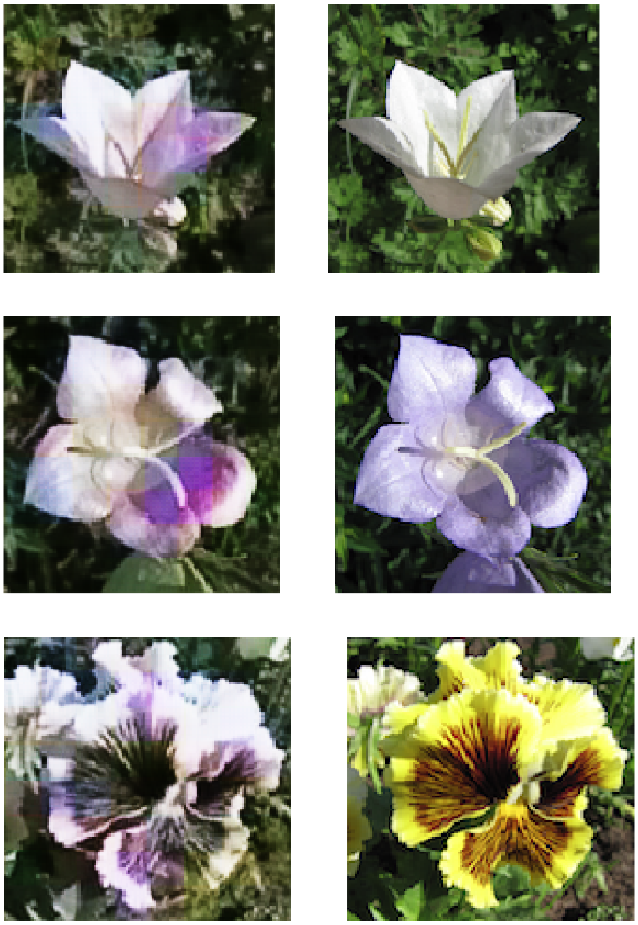


Fig. 16, 17 18 For each of the images we show the reconstructed image (left),
the original image (right)

From the images we can see that the reproduction of the validation set had a bias towards purple, possibly due to our dataset. The grayscale conversions were, however, plausible with little blurriness. In order to test the generalization further we randomly picked images from a google-search of flowers as the test set and ran them through our model, with results seen below.

Flower Images on Google:

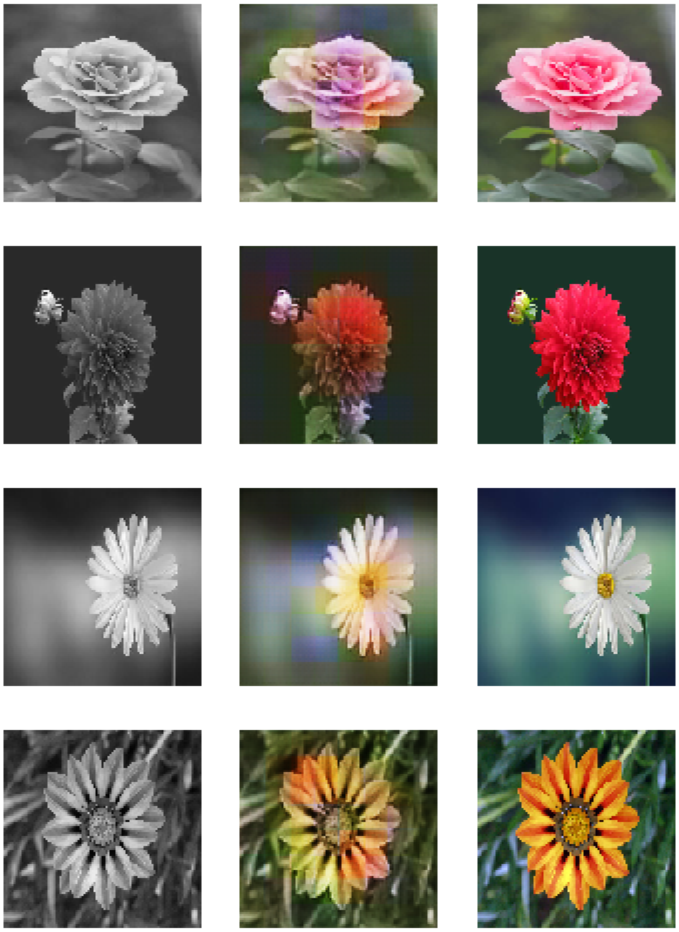


Fig. 19, 20 21
For each of the images from Google we show the grayscale (left), the reconstructed colored image (center) and the original image RGB (right)

Similar to the images from our validation set the results are plausible. Interestingly, the purple bias is only seen for the pink rose with the other three images reproduced almost perfectly.

Bird Images on Google:

Since there were no other images except flowers in our dataset, we wanted to test the capabilities of our model at coloring other pictures so we ran two images of birds through it.

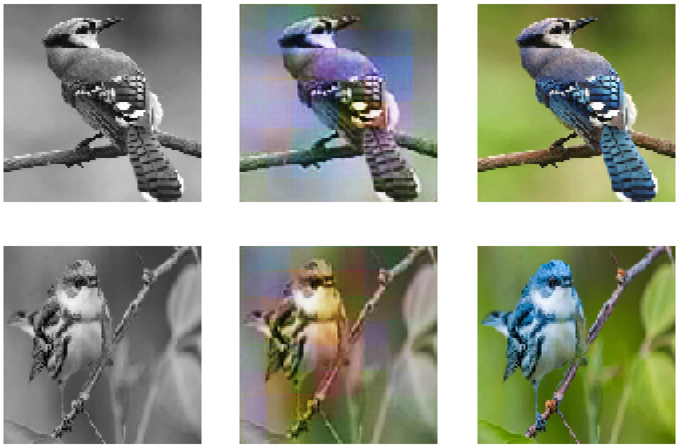


Fig. 22, 23
For the two bird images we show the grayscale (left), the reconstructed colored image (center) and the original RGB image (right)

Despite not being trained to color bird images, the model does a decent job at determining where the bird is and assigning its color. The reconstruction is far from perfect but that is to be expected given our model was trained only on flowers.

Discussion:

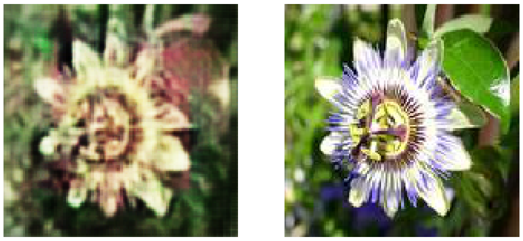


Fig 24. Result of Baseline Model

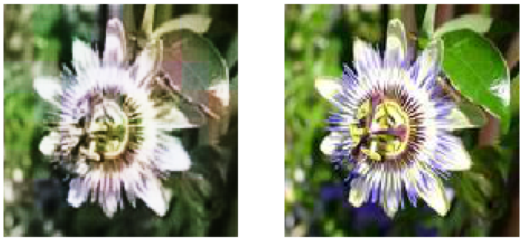


Fig 25. Result of Final Model

As displayed, the main improvements achieved by the final model are higher resolutions and more plausible coloring. As the team iterates through different GAN architectures and regularization strategies, the discriminator always end up dominating the model with perfect prediction accuracy. Refinement and tuning of the model is done with a three-pronged approach. The first pertains to the model architecture. The layers were increased in the generator to have more trainable parameters and feature maps. different types of activation functions were experimented such as sigmoid, relu, and leaky-relu functions. The ResNet style skip connections were implemented to negate the vanishing gradient. The second method regularizes the discriminator through conditional training based on its accuracy, implementation of drop-out layers, variation in learning rate, and the occasional noise input. The final approach was a custom loss value that sums the L1 loss and BCELoss in order to measure the pixel-by-pixel differences and accelerate training.

Aside from crippling the discriminator, the biggest noticeable change came from implementing skip-connections in the generator in order to traverse information more effectively and reduce the effects of a vanishing gradient. This proved to be most-useful in generating realistic images.

The biggest advantage that GAN have over the autoencoder is that there is no MSE loss when backpropagating, which sacrifices sharpness of image for matching colors. The resulting improvement in the final model was that it ensured that colours were relatively bright and images were sharp. This was contrary to the initial autoencoders that were developed(Fig 24, 25) in which despite efforts tuning hyperparameters the image quality was still blurry with unclear edges and square-like colour blocks.

The training set includes 100+ categories of flowers. Through quantitative and qualitative assessments, the model performed relatively well. In regards to the validation of 210 images, for most cases it correctly identifies where the main areas of colour for the flowers are versus the main areas of surrounding green backgrounds. As for new types of images, the model is less successful at realistically colourizing them, as seen by the above bird images in Figure 22,23. A notable mention is that we tried to train the model on 8000 completely random images, but the model failed to add any color. There are two likely explanations, the model may not have enough parameters to generalize or there are not enough repeated features in the training set since it is too diverse.

Ethical Considerations

The main limitation of the model is the training data, is that it only comprises flower images and as a result, performs poorly with other images. Furthermore, some more specific biases in the data include that most of the flower images were centered with surrounding green, leafy backgrounds, and the majority of the images were of coloured flowers. The result is that photographs in which there are non-green flowers in the background are more likely to be coloured unrealistically (in which the flowers will be coloured to look like leaves). The training dataset had a larger proportion of pink/purple flowers such as petunias, and the bias is thusly reflected most particularly in white flowers as seen in Figure 20, in which the model tends to give white flowers a pink-ish orange-ish hue.

Although this has no immediate dangerous ethical implications as flowers are a relatively harmless dataset, as this GAN was developed to be used with any type of data, it's worthy to note the importance of balanced and unbiased data, as mistakes or generalizations of colours could cause confusion of offense to viewers of images (e.g. a flower species that can only be one colour gets colourized as something different).

Project Difficulties

Notable difficulty of the project lies in tuning of the GAN. Due to high sensitivity and many hyperparameters, it became an arduous, time-consuming task. Since neural networks try to find the most efficient path to satisfy the training, the GAN was very sensitive to small changes in the network. The loss and accuracy of the model are not clearly indicative of the model's performance so the team had to be meticulous in tuning hyperparameters. Furthermore, handling the interaction between the discriminator and generator proved to be quite challenging as the discriminator had the tendency to overtake the generator; allowing neither to learn. The team tried many implementations of different GANs in order to extend the min-max optimization, by simplifying the discriminator architecture, modifying the training function, and increasing the complexity of the GAN in hopes of allowing it to train faster. In the end, the min-max game between the discriminator and generator was still not optimal, as towards the ending epochs of training the discriminator was still able to overpower the generator. Therefore, despite a relatively simply dataset of only flowers, the GAN colourization still produces certain outputs that may be deemed unrealistic. The project could be expanded by incorporating different skip connection methods e.g., DenseNet and HighwayNet or modified to be a conditional GAN. Although there is no certainty on the modifications' effects on performance, they could be some interesting projects for the future.

Google Collab Link:

https://colab.research.google.com/drive/1f4qeYdaWhYc8CCQJy13F6e9frDgXpR0Y

References

[1]A. Touchette, "A Quick History of Color Photography (for Photographers)", *Photo & Video Envato Tuts+*, 2019. [Online]. Available: https://photography.tutsplus.com/articles/the-reception-of-color-photography-a-brief-history--cms-28333. [Accessed: 14- Nov- 2019].

[2]D. Djudjic, “This is how you know which colors to add when colorizing black and white photos,” *DIY Photography*, 09-Jun-2017. [Online]. Available: https://www.diyphotography.net/know-colors-add-colorizing-black-white-photos/. [Accessed: 05-Dec-2019].

[3]R. Zhang, P. Isola, and A. A. Efros, “Colorful Image Colorization,” Computer Vision – ECCV 2016 Lecture Notes in Computer Science, pp. 649–666, Oct. 2016.

[4] K. Nazeri, E. Ng, and M. Ebrahimi, “Image Colorization Using Generative Adversarial Networks,” Articulated Motion and Deformable Objects Lecture Notes in Computer Science, pp. 85–94, May 2018.

[5]“Flower Datasets,” Visual Geometry Group - University of Oxford. [Online]. Available: https://www.robots.ox.ac.uk/~vgg/data/flowers/. [Accessed: 05-Dec-2019].

[6] O. Belitskaya, “Flower Color Images,” Kaggle, 31-Oct-2017. [Online]. Available: https://www.kaggle.com/olgabelitskaya/flower-color-images. [Accessed: 05-Dec-2019].

[7]
Preserve Knowledge, 25-Aug-2017, How to train a GAN, NIPS 2016 | Soumith Chintala, Facebook AI Research [Video file].
Available:https://www.youtube.com/watch?v=myGAju4L7O8 .[Accessed: 05-Dec-2019]