Project Final Report

Shu Zhang & Yueqi Mao

1. Introduction

Back to about February, we watched a video that demonstrated the process of making a short fan-made anime and were shocked at the amount of work required: a team of five fans devotes all their free time in one month to produce only hundreds of images for that simple animation. And a formal one requires even higher Frames per second (FPS), higher image quality, and longer length. The workload is huge: excluding the repetitive opening and ending, one episode of a traditional 2D anime is about 20 minutes, and these 20 minutes include thousands of frames. After finishing all the necessary preparation steps, including original project design, storyboard design, layouts drawing, and key animations drawing, the production staffs then need to colorize thousands of key animation sketches.

Considering that the colorization task is tedious, we quickly had the idea that it would be nice to figure out a way of automatic anime image colorization. After a few research, we chose to focus our topic on automatic image colorization using machine learning algorithms. Our hope is to colorize grayscale images, key animation sketches which contain more complex luminance information than grayscale images, or even black-and-white outline images completely without human intervention, but that would be too broad a topic. Therefore, we limited our topic to Convolutional Neural Network (CNN) based automatic image colorization for grayscale animation images with a few extensions to Generative Adversarial Network (GAN) models and to colorizing manga (black-and-white outline) images. The dataset we used was collected from a Japanese anime we watched before, called SK8 Infinity. We collected about 6000 images in our dataset, which could be useful for future research in this field. We will introduce the basic image colorization idea, the models we used, and the result we gained with necessary discussions in the following sections.

2. Problem Statement

As previously stated, it is difficult to colorize black-and-white outline images or exact key animation sketches, so we started from colorizing grayscale anime screenshots which, in addition to black-and-white outline information, contains the luminance information in images. Our goal is to research the algorithms for automatic grayscale image colorization using CNN based models and evaluate their performance on our own datasets. While there is no existing perfect model for this goal, we will make analysis, make modifications to the models, and try to identify problems as future research directions.

Our input and output are as follows:
CNN:

- input: single channel grayscale image, which provides Y value, representing the luminance information of the image in YUV color space
- output: predicted U and V values, representing the color information of the image in YUV color space

GNN:

- input (for the generator): grayscale image (R=G=B), only containing luminance information with range 0 – 255
- output: fake RGB image

3. Data Collection

The first approach we used for data collection was taking screenshots of the anime manually. We took 48 screenshots of episode 1, and 56 screenshots of episode 2, and used them for our first training attempt. In this stage, all our images of the dataset are selected carefully. To ensure the training result, we selected images in the same light (e.g., daylight), containing characters dressing consistently (e.g., always wearing a black hoodie). But soon after we realized that we wanted the dataset to be larger and more comprehensive, we switched to the second approach.

The second approach is to take automated screenshots. We use VLC Media Player to achieve this goal. We set up the scene filter in VLC and let it take screenshots for us automatically, our dataset size expanded from 100-500 to 6000. After several adjustments, we let the VLC take one screenshot per 60 frames to get distinct images (screenshots) without duplication. And to avoid using too much RAM and to save training time, we compressed our image to approximately 256 x 160 pixels.

4. Algorithm Architecture:

Based on our research of automatic image colorization methods, we have decided to start using a convolutional neural network (CNN) based algorithm. The model we chose is built on the paper *Fully automatic image colorization based on Convolutional Neural Network* by Domonkos Varga, et al. [1], and our code is based on a simplified online version [2] that is constructed using the same paper. We will give a necessary explanation of the neural network backgrounds, typical CNN model architecture, and the model that is used for our grayscale image colorization algorithm, in this section.

4.1 Neural Network Background

Neural network, by its name, refers to the collections of neurons connected as a complex network. It is a mimic of a brain network that consists of millions of nerve cells. While our brain can receive information and make decisions, the neural network model aims for the same goal but implemented on a machine. Each artificial neuron represents a mathematical operation that can receive signal from the previous neuron and make calculation according to a given function with weights and bias. If the calculated result passes a threshold, then the neuron triggers a signal and passes it to the next connected neuron, if not, it stays inactive. Since the number of neurons are huge in a network, they are separated into different layers, and the last layer would be used for calculating the result. All the signals are summed up using a non-linear function and the result is compared to the expected output, which is usually extracted from the training data. The difference, which is called the error and calculated using a loss function, will be used in the next training iteration. The model keeps learning and updating in the process and the loss value decreases while the accuracy increases. Eventually, the model should be trained to perform as closely to the training data as possible.

4.2 CNN Model Architecture

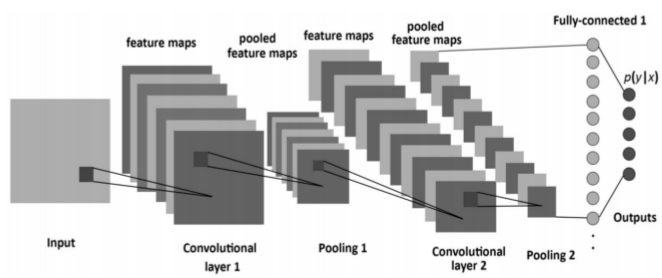

Figure 1. Basic feature-based CNN model architecture [3].

As an extension to basic neuron network, a CNN model contains a few additional layers that are specific to the idea of convolutional. Simple neuron nets, for example, a fully connected neuron network, may easily encounter overfit problems where there are too many neurons for trivial variables and the model loses representative for general cases. Thus, a CNN model adds an additional layer for extracting features from the inputs and transforming them into a simplified representation map using less variables. And when producing the output, it uses a reverse strategy and transforms the feature-based representation back to the normal inputs.

4.3 CNN Model for Grayscale Image Colorization

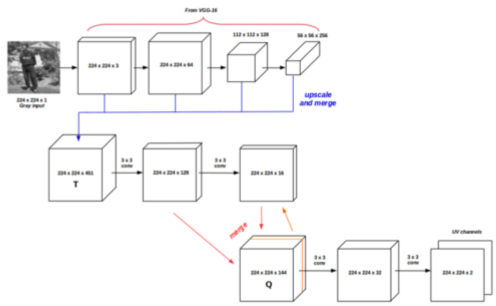

Figure 2. The CNN model proposed for grayscale image colorization [1].

The model we used, based on the paper we read, uses a two-stage feature-based CNN architecture called VGG-16. Despite the specific modifications made to prepare the model for image colorization purpose, it is fundamentally just a CNN model that can make U, V channels predictions. Given the input RGB training data, the algorithm first converts them into single channel grayscale images and then passes them into the CNN model to make U, V channel predictions. While the process involves using complex matrix calculation operations and loss functions, we will skip the details and direct further readings to the paper [1].

4.4 GAN Model Architecture

Despite the effort we devoted in minimizing the overfit problems in CNN model, the result is not satisfying. We will discuss more in the result section, but we want to first introduce our second approach here, which is using a GAN model.

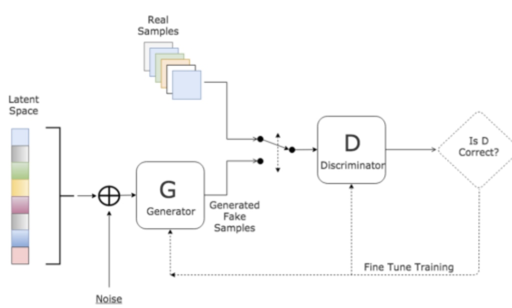

Figure 3. A basic GAN model structure [4].

The idea of GAN, again from its name, is to let two neural networks compete. That is saying, it typically contains two components, one called generator (G) and one called discriminator (D). The training process begins with first letting the generator generate completely random fake output, and then pass the fake data along with real data to the discriminator and let it distinguish between which one is real and which is fake. The generator tries to fool the discriminator while the discriminator aims to always identify the real ones. In the end, both should achieve satisfactory performance and the generator can produce realistic images.

4.5 GAN Model for Colorizing Grayscale Anime Image

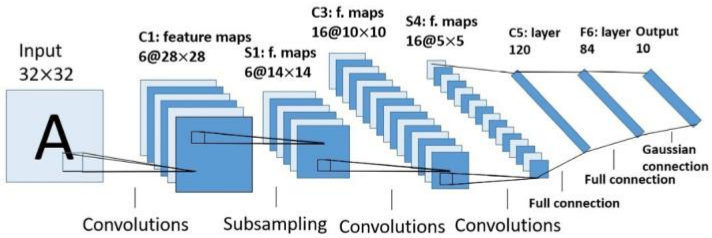

Figure 4. The GAN model proposed for grayscale anime image colorization [5].

The GAN model we used is based on the paper *Colorization of anime gray images via generative adversarial networks* by Xinyu He, et al. [5], and the model is again an extension to the basic GAN model. The major modifications here are made for optimizing the image colorization tasks. Here, the generator is an improved CNN network called U-Net, where the model is better at learning both global and local characteristics of an image using a structure called non-local blocks. The discriminator is another variant of CNN network called LeNet, which is a 5-layer CNN that contains the convolutional layer, pooling layer, and the fully connected layer. The structure is not complex, but it gains all the advantages of a typical CNN network [5]. Again, further readings could be found in the paper [5] where more mathematical formulas and details are explained. Our model is a simplified version of this network with reference to online codebase [6].

5. Result

We started from training the CNN model on about 400 training images and 100 testing images for 200 epochs, then increased the size to 4500 training and 1500 testing for 500 iterations. However, the result did not change much, and we are not satisfied. Even for the training datasets that the model has already seen, the accuracy statistics are not bad, but the result is wired.

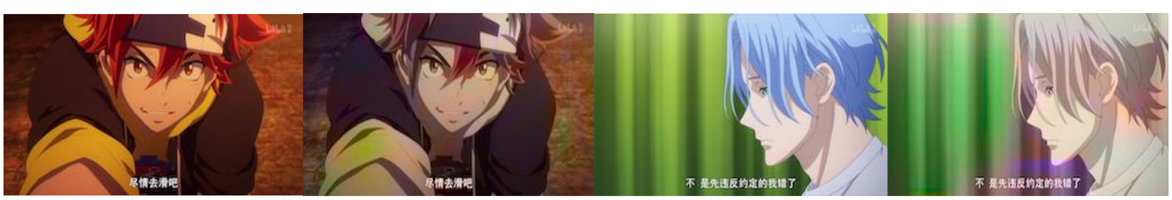

Figure 5. Example of training data compared with real data (the model has already seen).

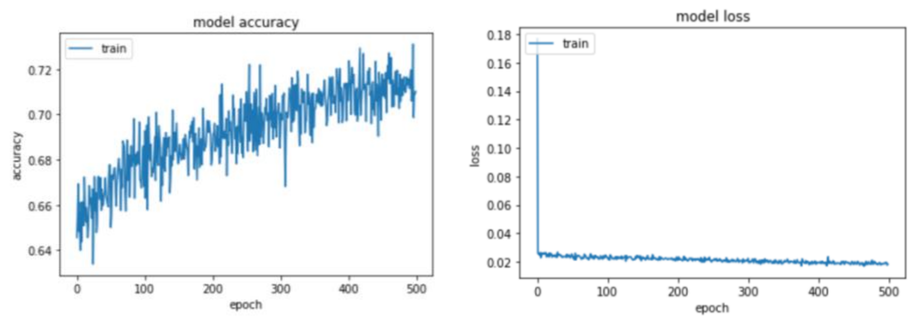

Figure 6. CNN model accuracy and loss statistics (not bad actually).

Therefore, we turned to GAN model and the training result for about 6000 randomly selected images (4500 for training and 1500 for validation) after about 300 epochs are much better. For the training sets, the fake images are the same as the real images, despite a few details and some minor blurring (like the eye colors or cloth decorations).

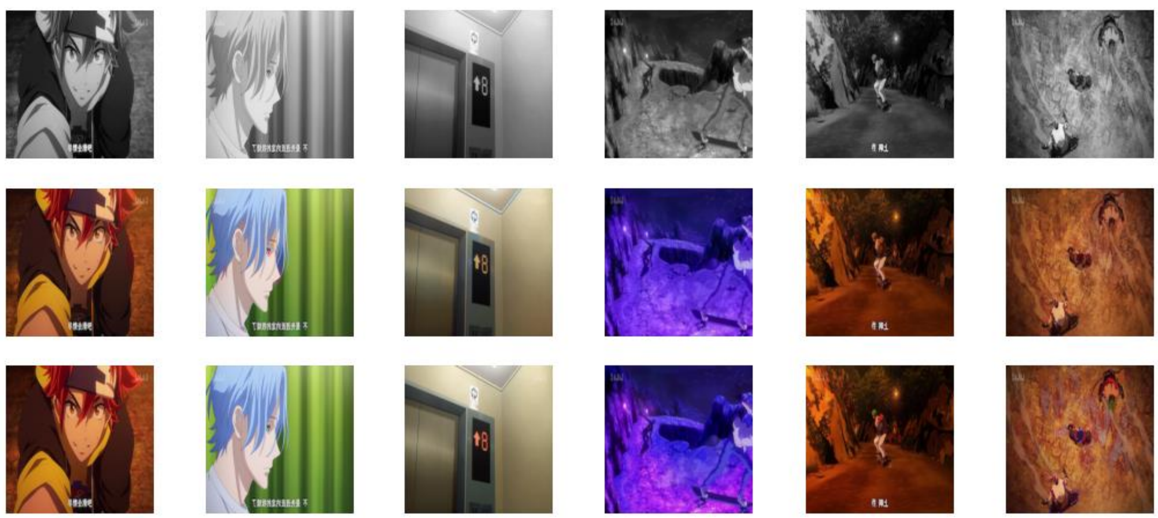
Figure 7. Example of training data compared with real data for GAN.

Here, you can clearly see that the results are much better than CNN model (the first two columns can be compared with the CNN result shown above in **Figure 5**). The first row is grayscale image, the second is the fake one generated by GAN, and the third row is the real one. Although all the images have been given to the model in the training process, it is still a significant improvement compared to the results from CNN. And if you take a closer look at the images, you will be able to find a lot of trivial differences like a “spot the difference game,” which is interesting.

Now comes the more interesting part. Although the GAN model can gain much better results for images it has already seen before, it performs bad for new images. Few improvements have been achieved compared to CNN models.

Figure 8. Example of testing data compared with real data using CNN and GAN.

All the images above are new which means the model has never seen them before. And you can see although GAN achieved a much better performance than CNN, it is still not acceptable as the results are wired to human perspectives. We also tried to colorize a few manga images, which are just black-and-white outline images with no grayscale information. The task is harder, and the result is again not so good but interesting.

Figure 9. Example of manga image colorization using CNN and GAN.

6. Analysis

There are a few interesting things we noticed from our results. First, both models have overfit problems although CNN is worse than GAN. GAN has an almost perfect performance with the training dataset, but still generates non-sense colorization for new images and for manga data.

Second, animation and manga image colorization face unique challenges compared to photographs. Not all current papers are focused on animations, where most of them use real photographs or movie scenes, so the current model may not fit the animation colorization well. And one example would be, the photographs catch real world scenes, and it follows the rule of nature. The sky is always blue, and sunshine is always bright. But animations often have their own logic, or the painter will try to exaggerate certain scenes following personal styles. All those irrational scenes would bring confusion to the model. Also, photographs use filters on top of original images while animation uses a completely different coloring palette for those scenes, for example, for a memory scene or a night scene. Finally, there are also good and interesting results here. As we discussed before, the GAN model can colorize according to its own logic. Although the result is not what we expected and cannot be used directly, it is not completely random. For example, it can identify the color of one of the main characters and tried to colorize it as red for most of the new animation images, even for one of the manga images. But of course, further research and investigation is needed to keep going down this track.

7. Conclusion & Future Direction

Our project aims for colorization grayscale images using CNN based models. We tried one for CNN and one for GAN, which is a combination with two CNN and trained in a competition. We can generate reliable performance for data in the training datasets using the GAN model trained on about 6000 random scenes after 300 iterations. However, the results of colorizing new animation images or manga images are not as good as expected. We identified a few challenges and potential issues causing this result including overfit problems and animation difference to movies and photographs. Although we did not get the time to fix them, we provide those as future research directions. There is also other CNN, GAN related models that are worth evaluations, including DCGAN, CGAN, and more, which we also recommended as future exploration potentials. That is saying, we are looking forward to the future of fully automatic animation or even manga image colorization when anime production teams can concentrate on making better stories.

8. Appendix

Here is our link to datasets & code in google drive (the datasets are too big to be uploaded to canvas or GitHub):
https://drive.google.com/drive/folders/1QMmluxQkMa0Mn0vIbfLIjaypuEGVUJx-?usp=sharing

Reference

[1] D. Varga and T. Sziranyi, "Fully automatic image colorization based on convolutional neural network," *IEEE Xplore*, Dec-2016. [Online]. Available: https://ieeexplore.ieee.org/document/7900208. [Accessed: 29-Apr-2022].

[2] Rrupeshh, "Rrupeshh/auto-colorization-of-grayscale-image: Automatic colorization of the grayscale images," *GitHub*. [Online]. Available: https://github.com/rrupeshh/Auto-Colorization-Of-GrayScale-Image. [Accessed: 29-Apr-2022].

[3] A. Shrivastav, "Different types of CNN Models," *OpenGenus IQ: Computing Expertise & Legacy*, 25-Nov-2021. [Online]. Available: https://iq.opengenus.org/different-types-of-cnn-models/. [Accessed: 29-Apr-2022].

[4] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," *arXiv.org*, 10-Jun-2014. [Online]. Available: https://arxiv.org/abs/1406.2661. [Accessed: 29-Apr-2022].

[5] H. Xinyu, Y. Lu, and Y. Yang, "Colorization of anime gray images via generative adversarial networks," *IEEE Xplore*. [Online]. Available: https://ieeexplore.ieee.org/document/9574551. [Accessed: 29-Apr-2022].

[6] "Image Colorization with U-Net and GAN Tutorial," *Google Colab*. [Online]. Available: https://colab.research.google.com/github/moein-shariatnia/Deep-Learning/blob/main/Image%20Colorization%20Tutorial/Image%20Colorization%20with%20U-Net%20and%20GAN%20Tutorial.ipynb#scrollTo=qKx2TL4_HpEd. [Accessed: 29-Apr-2022].