Colorizing Black and White images using CNNs

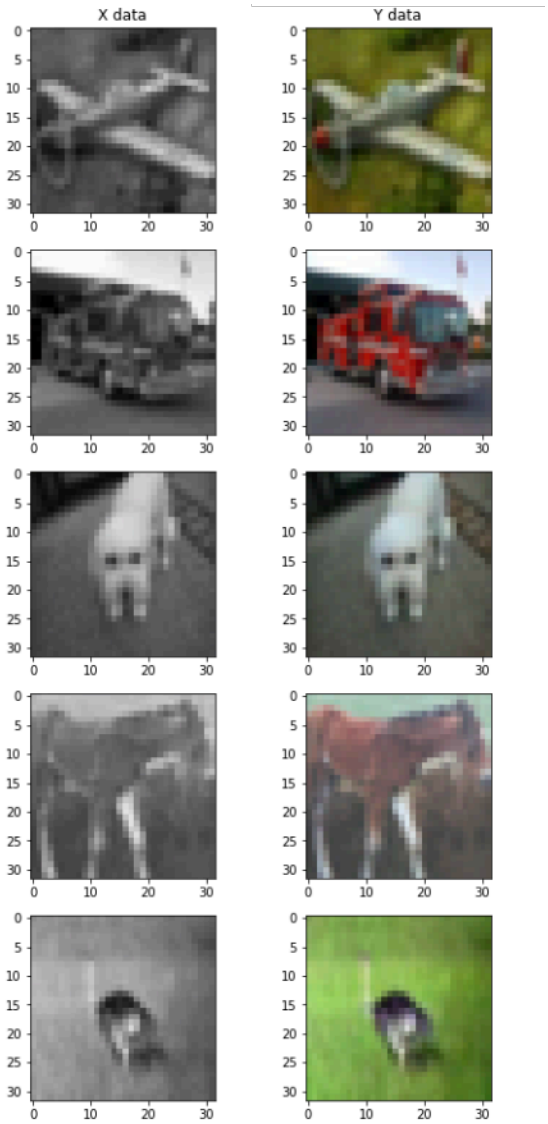
Alexander Bobronnikov
University of Massachusetts Amherst
abobronnikov@umass.eud

1. Introduction

For this project, I wanted to focus on a vision/image-based machine learning problem as I found these types of uses of machine learning very interesting during our lectures. After doing research into common image-based machine learning problems, I decided to try to tackle the problem of colorizing black and white images. This was a very interesting problem area due to how applicable and useful it is in the real world. There are thousands of black and white images from before colored cameras were invented, that can be brought to life by simply adding color. To approach this problem, I am planning on creating a supervised neural network that gets fed black and white images and outputs the colored version. I will lay out my plan to do this in more detail in the technical approach section.

1.1. Problem Statement

As stated in the introduction, the problem I am working on for this project is creating a neural network that is able to accurately colorize black and white images. To do this, I am using the CIFAR-10 dataset. There are pros and cons to using this dataset as the images provided are only 32x32. An advantage of this is that training will be much more efficient than if I were to use higher resolution images. A disadvantage of using these smaller resolution images is that there will be much less context for the network to be able to determine what is happening in the image which will reduce the networks ability to accurately colorize these images. For this reason among others, I do not expect the results to be perfect. I expect this network to be able to generate images that will look like the original grayscale image with plausible color. To this point, I believe a good evaluation standard for these images would be whether the generated images look like a plausible colorized version of the original grayscale image. Below I will display some of the grayscale images and colored images randomly picked from the dataset.



1.2. Technical Approach

The initial approach I am planning to use for this problem is to use a multi-layer convolutional network with batch-normalization to extract features from the image. Additionally, I plan on trying to add a pass through of the original image to the final layer, so the network is better able to keep the structure of the image and use the learned features from the convolutional layers to determine how to color this structure. Another possible approach I read about in regard to generating images from neural networks would be to use an autoencoder structure. This structure could be helpful because the convolutional layers reduce the number of features in the network. The autoencoder helps with this problem by up sampling to get back to the original image size after the convolutional layers learn the features. For the loss function, I am currently using the mean squared error between the predicted colored image and the actual colored image, however I may try to create a more task specific loss function that takes into account how smooth the image looks and the actual structure of the image being maintained.

1.3. Preliminary Results

At this point in the project, I have done the preprocessing of data and developed a basic model to compare my results against. Because the CIFAR-10 dataset contains colorized images, I had to convert these images to grayscale to use as training data. To do this, I used the formula to calculate the luminescence of each pixel: 0.299 * r + 0.587 * g + 0.114 * b. For my basic model, I created a network with just one fully connected layer (3072x3072) to use as a control group to compare my final models against. Obviously these results should not be accurate at all as this neural network is essentially just a linear regression because there is only one layer. The preliminary results of this basic network after training for 50 epochs, show that, as expected, this network does pretty poorly. While this network is able to determine fuzzy darker and lighter errors corresponding to the grayscale image, the full structure of the images is completely lost with this basic model and the colors produced are nowhere near accurate and not smoothly distributed. Below are 10 random examples of predicted images compared with the grayscale input as well as the actual colored image.