Arnaud Gardille

1 November 2021

Report on image colorization

The task of colorising greyscale images previously needed significant user inputs but can nowadays be automated thanks to the recent improvement of deep learning.

The objective of this challenge is to colour images given grayscale input image. The dataset contains 4282 images in JPEG format.

My first approach was simply to work in the RGB colour space. It worked, and the learning process was quite fast. However, the loss of information when passing through smaller layers led to blurred images.

After some research, I found an article with an interesting approach consisting in working in the LAB colour space. It consists in separating the luminance L from the colour components A and B and is intended to be perceptually uniform. Then, our model only must learn how to produce those A and B components and adding them to the greyscale image. Even if the colourisation is still not so precise, the final image now looks sharp enough.

The model implemented remains unchanged, except for the fact that it output a 2-dimensional AB image rather that a 3-dimensional RBG image.

DataLoader

Regarding pre-processing, I do some data augmentation through the RandomCrop and RandomHorizontalFlip transforms. In the _getitem_ fonction, I must decompose the original image in the LAB colour space.

Optimizer

I use the Adam optimizer, which is a widely used improvement of the SGD.

Metric

I chose to use the MSE loss, it is to say the L2 norm, as it is the standard for image comparison. I also tried the L1 norm, which gives more colourful results, but with more mistakes.

I have to precise that the loss is computed on the difference between the AB images, and not the RGB images.

Inference script
The file predict_color.py can be used to predict the color of a greyscale image using my model.

```
python predict_color.py -image example_gray.jpg -model trainedModel
```

Improve the image quality of the model
The use of the LAB colour space improves the quality of the result, without significant changes in the model. Another possibility would be to use the predicted colour of the RGB model, and to adjust it to fit le lightness of the input.

Average mood of the image
The function get_histogram in train.py produces histograms for the three colour channels. An example is given below. You can access the histogram of the generated image within the inference script.

```
python predict_color.py -image example_gray.jpg -model trainedModel
-histo True
```

Sources
https://lukemelas.github.io/image-colorization.html

https://towardsdatascience.com/histograms-in-image-processing-with-skimage-python-be5938962935