RainbowNet: Color Extrapolation from Grayscale Images

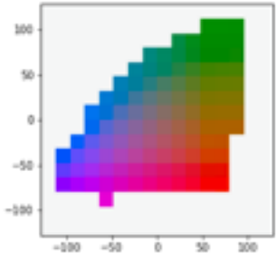
Cindy Lin, Emily Ling, Owen Wang | {cinlin, eling8, ojwang}@stanford.edu | CS 229 | Stanford University

Summary

Inspired by digital recoloring of historical photos and Zhang et al.[1], our project aims to train a convolutional neural network to colorize black-and-white images, augmenting with embeddings obtained from Google's Inception ResNet v2[2]. Given a batch of RGB images, we generate a possible colorization by training and predicting in the CIE Lab color space. We additionally analyze our training data to examine which distributions of colors often occur together via K-clustering. Our model predicted majority brown colors and did not perform well with the added complexity of the IRN2 fusion.

Features

Our features consist of two components:
1) L channel of raw image input obtained via preprocessing (see Data section) with dimensions (224, 224, 1)
2) ResNet v2 embeddings with dimensions (1000,) for image classification, used here to provide more semantic information of the image to our model



Since loss functions that minimize the distance between the predicted and the true label produce desaturated images, we instead use a bucketing schema which quantizes the ab output space and maps them to valid values in RGB. We thus output 112 buckets, illustrated in the figure to the left.

Model

Our model has three main components:
1) Encoder: 8 convolutional and 4 normalization layers
2) Fusion (optional): combination of encoder output and ResNet embeddings as described in Iizuka and Simo-Serra et al.[4]
3) Decoder: 12 convolutional and 3 normalization layers

Output: per-pixel softmax prediction over 112 ab buckets
Loss: cross-entropy
The function for convolutional layers per-pixel is written as:

$$y_{x,y} = \phi\left(b + \sum_{i=-k_x^L}^{k_x^L} \sum_{j=-k_y^L}^{k_y^L} W_{k_y^L+i,k_x^L+j}\, x_{x+i,y+j}\right),$$

$$k_x^L = \frac{k_x - 1}{2}, \qquad k_y^L = \frac{k_y - 1}{2},$$

W is a shared matrix of weights, k's are the kernel height and width, and x, y are the pixel component of the input, output.

Data

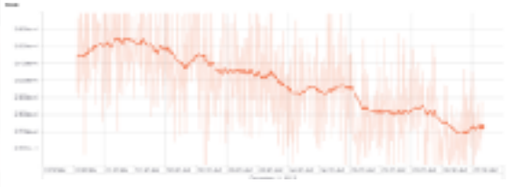
Our dataset contains 10K images from Unsplash[3] that are mostly 256x256 portraits and are preprocessed as follows
1) RGB images resized to 224x224
2) Feed grayscale images into ResNet to obtain embeddings
3) RGB images normalized to [0, 1] and converted to Lab color
4) Lab images separated into L and ab channels
5) L channel normalized to [0, 1]
6) ab channels discretized into 112 buckets

Training data: L channel, ResNet embeddings
Ground truth: ab channels

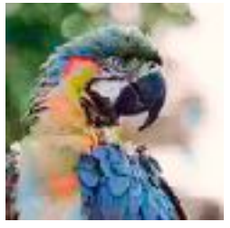
Results

- Proof-of-concept: was able to overfit on a tiny dataset (Fig. 1)
- Experimented with and without IRN2 embedding fusion: using fusion dramatically slowed learning
- Additional training would improve results; validation and train loss both showed signs of further decreasing (ep. 18-23 below in Fig. 2)

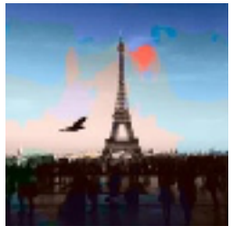
Figure 1, left. Figure 2, below.



- Below: Selected predictions on the training set, demonstrating our model was able to learn

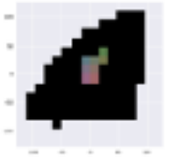


- Below: Selected predictions on the test set



Discussion

To better visualize our data, we used K-clustering on our training set, representing each image by a histogram of the number of times each color bucket appears, then averaging the counts per clusters. Some clusters are visualized below, with all buckets with non-zero frequency filled with color.



- Training time: our max training time was ~12 hours, after which the loss is around 2.5M; perhaps longer training?
- IRN2: fusion with ResNet embeddings might have been too complex for our model, thus producing poor results; plot on the right demonstrates validation loss per epoch
- Sepia: the majority of the model's color predictions were brown, since it is the most prevalent color in our training set

Future

1) Color weighting: to dissuade the model from predicting neutral colors, we can weight different colors in our pixel-wise cross entropy loss such that predicting rare, saturated colors are penalized less than predicting neutral colors, as inspired by Zhang et al.[1].
2) Sequential training: to colorize a sequence of images, or videos, we hypothesize that feeding in stacks of n consecutive frames into the model would produce a more stable colorization than frame-by-frame and would be more efficient than a RNN-CNN alternative.

References

[1] R. Zhang, P. Isola, and A. A. Efros, "Colorful Image Colorization," Computer Vision – ECCV 2016 Lecture Notes in Computer Science, pp. 649-666, 2016.
[2] "Improving Inception and Image Classification in TensorFlow," Research Blog, 31-Aug-2016. [Online]. Available: https://research.googleblog.com/2016/08/improving-inception-and-image.html.
[3] FloydHub - Deep Learning Platform - Cloud GPU. [Online]. Avai le: https://www.floydhub.com/emilwallner/datasets/colornet.
[4] S. Iizuka and E. Simo-Serra, "Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification." [Online]. Available: http://hi.cs.waseda.ac.jp/~iizuka/projects/colorization/data/colorization_sig2016.pdf