Machine Learning

Texture for Colors: Natural Representations of Colors Using Variable Bit-Depth Textures

Vsevolod Avilkin

Svetlana Pavlova

Mikhail Fedorov

Victor Adamovich

Artem Vergazov

Motivation

New efficient way of encoding an image

Usually autoencoders decrease the spatial size of the input image and encode it as a vector. One can consider an alternative - reduce the number of bits used to encode the image, in the extreme case use only 0 or 1 as encoding for every pixel. This also leads to perceptually pleasant binary images.

Problem statement

Many methods have been proposed to convert color and grayscale images to their binary counterparts with one bit per pixel. Typically, the goal is to improve certain attributes of the original image to make it more suitable for analysis. However, when the resulting binary image is intended for human viewing, aesthetics must also be considered. Binarization techniques such as halftone, shading, and shading have been widely used to model the intensity profile of the original image.

Related work

The closest work to ours in terms of motivation is Color2Gray. Color2Gray attempted to handle the isoluminant variations that were not preserved with standard color to gray-scale conversions through explicitly analyzing chrominance and luminance differences. We handle isoluminance; however, we employ an information preservation perspective and dramatically reduce bitrates. Similar problems have arisen in numerous specialized applications that require converting photographs to stylized bi-tonal renderings.

Neural model

For prepossessing, we apply linear normalization to the data to convert the original images to the tensors with zero mean and unitary standard deviation. It is an important step that helps stabilize the learning process. It is omitted in the original paper, so we decided to do it as an experiment.

The encoder part of the work is to reduce the dimensionality of an input image by passing the input through a series of non-linear transformations, including a low-dimensional “bottleneck” layer. The decoder part of the algorithm recreates the original image from the internal compressed representation emitted by the bottleneck.

Conducted experiments

For our experiment, we implemented the neural network in PyTorch. We trained our model on the STL-10 dataset which contains 100000 unlabeled images. We chose this dataset instead of well-known CIFAR-10 in order to make use of the fact that the algorithm is unsupervised and no labels are required for training. We trained our model on the STL-10 dataset which contains 100000 unlabeled images.

Results

As a result a neural network model, which replaces an image with its binary texture pattern and reproduces the original image from it was presented was performed.