Context

Given 2 datasets of palettes taken under 2 conditions (light on/light off) and pictured in different angles (left, right, ..). We want to find out whether those 2 datasets are sharing the same structure of representation.

Approach

Embedding images into a latent space (compressed representation) by the method of autoencoders. The key idea is to look at reconstruction error to measure how likely the structure of palette images comes from the same distribution.

Formal definition

Given a training dataset $\mathbf{X}$, for each sample $\mathbf{x} \in \mathbf{X}$, we define:

- encoder function $\mathbf{f}$, map $\mathbf{x}$ to other space $\mathbf{h = f(x)}$.
- decoder function $\mathbf{g}$, reconstruct $\mathbf{h}$ to its original information, $\mathbf{r = g(h)}$
- reconstruction error is calculate by loss function $\mathbf{L(x, r) = L(x, g(f(x)))}$

The first step is to learn the latent space $\mathbf{h}$ of each dataset. $\mathbf{h}$ is expected to contain useful properties of images. We assume that each dataset can be fitted well into its latent space (it means there are fewer outlier images represented in the dataset).
→ first step: verify this assumption by training 2 separated autoencoders for each dataset.

As a result of 2 separated training, we have:

- $\mathbf{h_1 = f_1(x)}$, for $\mathbf{x} \in \mathbf{X_1}$
- $\mathbf{h_2 = f_2(x)}$, for $\mathbf{x} \in \mathbf{X_2}$

We do not care about the decoder part $\mathbf{g}$ as long as it gives a low or an acceptable reconstruction error.

If 2 datasets $\mathbf{X_1}$ and $\mathbf{X_2}$ are sharing the same structure, my assumption is:

- If we merge those 2 datasets, they can be fitted well in the same latent space
  → second step: train an autoencoder with a merged dataset.
- If we take an image $\mathbf{x_1}$ from dataset $\mathbf{X_1}$, encode it into latent space $\mathbf{h_2}$. Is this possible to reconstruct $\mathbf{x_1}$ by the learned components $\mathbf{h_2}$ and $\mathbf{g_2}$?
  → third step: calculate the loss $\mathbf{L(x_1, g_2(f_2(x_1)))}$ and $L(\mathbf{x_2}, \mathbf{g_1}(\mathbf{f_1(x_2)}))$

Image processing

Since our datasets contain high-quality images with large dimensions, it poses a big challenge to fit any machine learning model. We experience different *reducing* techniques.

Firstly, from a human perspective, a coloured image looks identical to its grey version. We convert our dataset to grayscale, it helps reduce the 3-channel images into a 1-channel.

Secondly, resize images into a smaller size. It can be done with both coloured versions and grey versions.

Thirdly, cropping different parts of an image with equal dimensions. Then feed the cropped images into our model.