Computer Vision 1: Homework 2

Important:
Mark the homeworks you solved in the homework sheet and bring your solutions with you to the exercise class. For each homework problem, one student will be chosen at random to present their solution.

Programming tasks.

Appearance based classification. Training:

- Download the CIFAR-10 dataset (Python version) at `https://www.cs.toronto.edu/~kriz/cifar.html`.
- Read the first data batch in file `data_batch_1` as instructed on the webpage.
- For each of the classes "automobile", "deer", and "ship" (labels 1, 4, and 8, respectively), extract the 30 first images.
- Calculate and store the grayscale histograms of all of these images. First convert the images to grayscale by the averaging method. That is, for every pixel, the grayscale value is calculated as $(R+G+B)/3$. Then calculate the histogram using a fixed set of 51 bins covering the range $0-255$, each bin has length 5. **Do not take the automatically chosen bins by numpy.histogram** (Why?).

Testing:

- Read the test batch in file `test_batch`. For the same classes as above, extract the 10 first images.
- Calculate the histograms of all the images in the same way as above.
- Classify each image in the test set by finding its nearest neighbour in the training set. For each test image:
  1. Calculate the Euclidean (L_2) distances of the histogram of the test image and the histogram of *every* training image. If you have n training images, you should calculate n distances for every test image.
  2. Classify the test image by finding the minimum of the distances. Once you find the minimum distance, the predicted class of the test image is the class of the nearest training image.
- Calculate the classification accuracy as the ratio of the number of correctly classified testing images and the total number of testing images.
- Feel free to change the bins to some other lengths like 1, 10 or 255, and see how the classification accuracy varies.

Hints:

- When you have loaded the data (in the dictionary called `train`), this is how you can access the color channels of the `i`'th image.

```
raw_data = train["data"][i,:]
red = raw_data[:1024].reshape((32,32))
green = raw_data[1024:2048].reshape((32,32))
blue = raw_data[2048:].reshape((32,32))
```

- Be careful with the data types. Do not directly sum `uint8` arrays, but first convert them to floating point to avoid overflow.

Other tasks.

1. Figure 1 shows a normalized histogram $p(X)$ of pixel intensity in a grayscale image.
   - Find the expected value of X, i.e., $\mathbb{E}[X]$.
   - Draw the cumulative histogram of X.

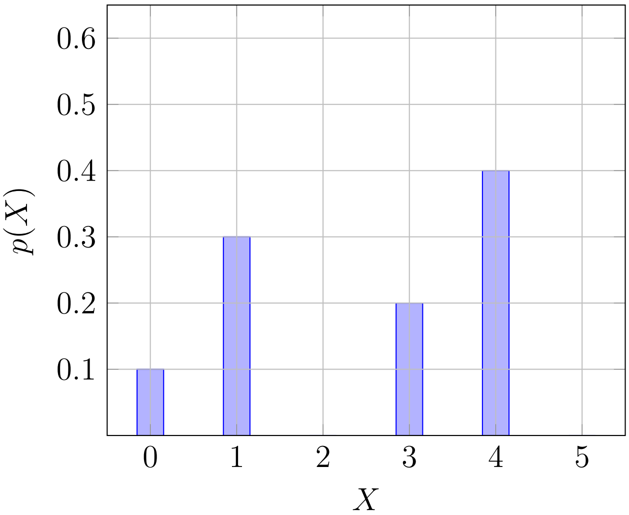

Figure 1: A normalized histogram $p(X)$.

2. Calculate the L_1 (Manhattan) distance between $p(X)$ and $q(X)$ (Figure 2).

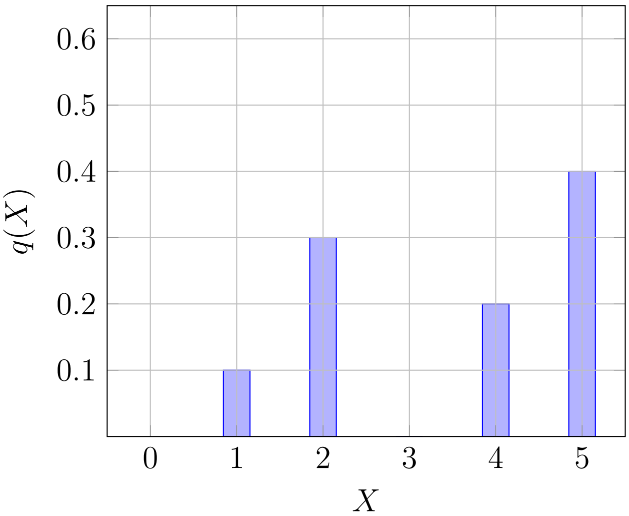

Figure 2: A normalized histogram $q(X)$.

3. Let X be a random variable denoting the intensity value of a pixel in a grayscale image. By the definition of variance,

$$\mathrm{Var}(X) = \mathbb{E}\left[(X-\mu)^2\right],$$

where $\mathbb{E}$ denotes expectation, and μ is the expected value of the random variable X, i.e., $\mu = \mathbb{E}[X]$.

Suppose you multiply the intensity value of each pixel by a scalar $a > 0$. Using the definition, prove that $\mathrm{Var}(aX) = a^2\mathrm{Var}(X)$.

Hints: Square of a binomial, linearity of expectation

Note: the definition of variance in this exercise is more general than the one in the lecture slides.