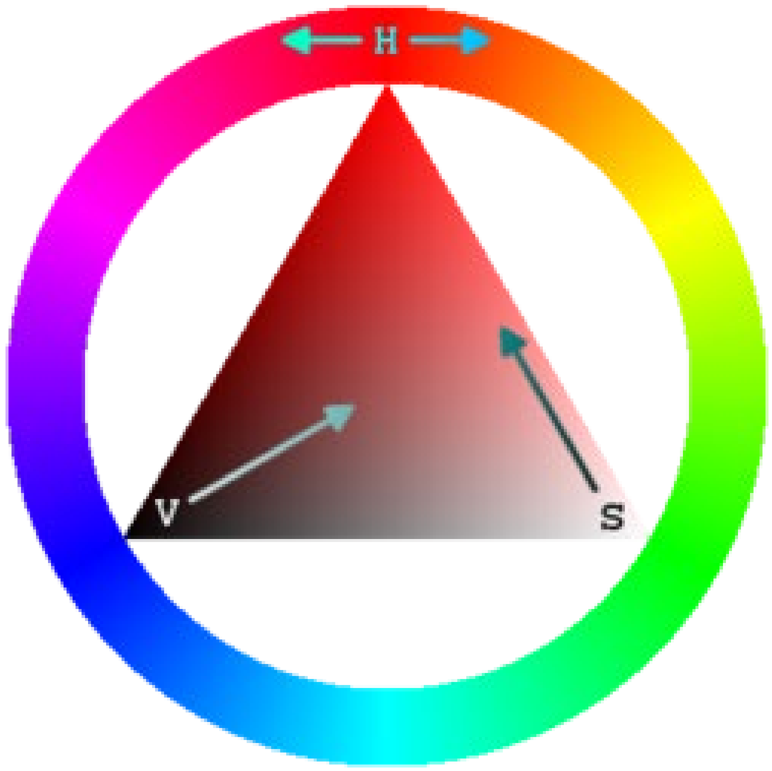
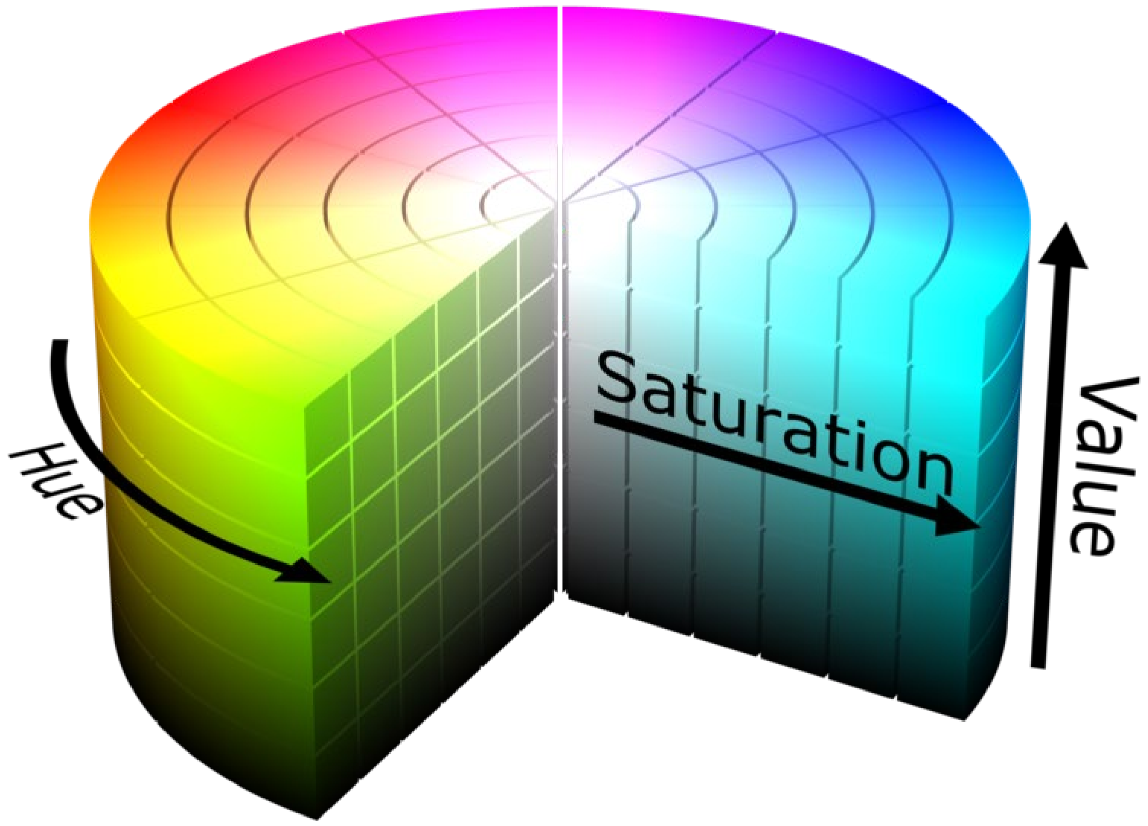
Extract Features

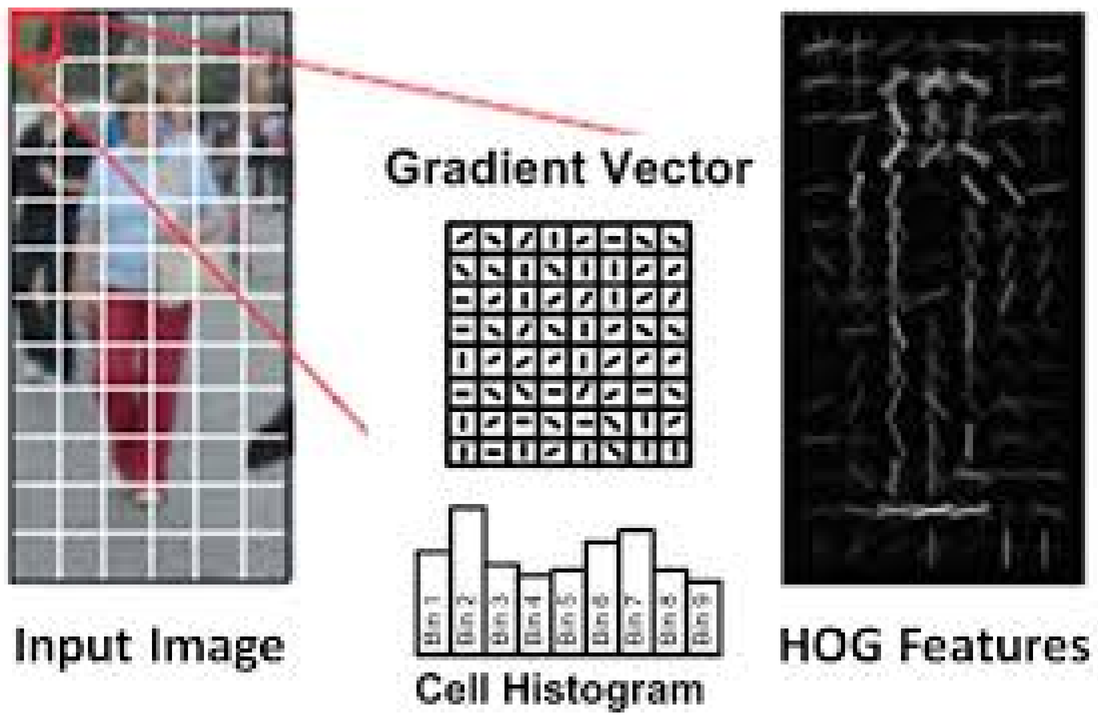
HSV Color Histogram

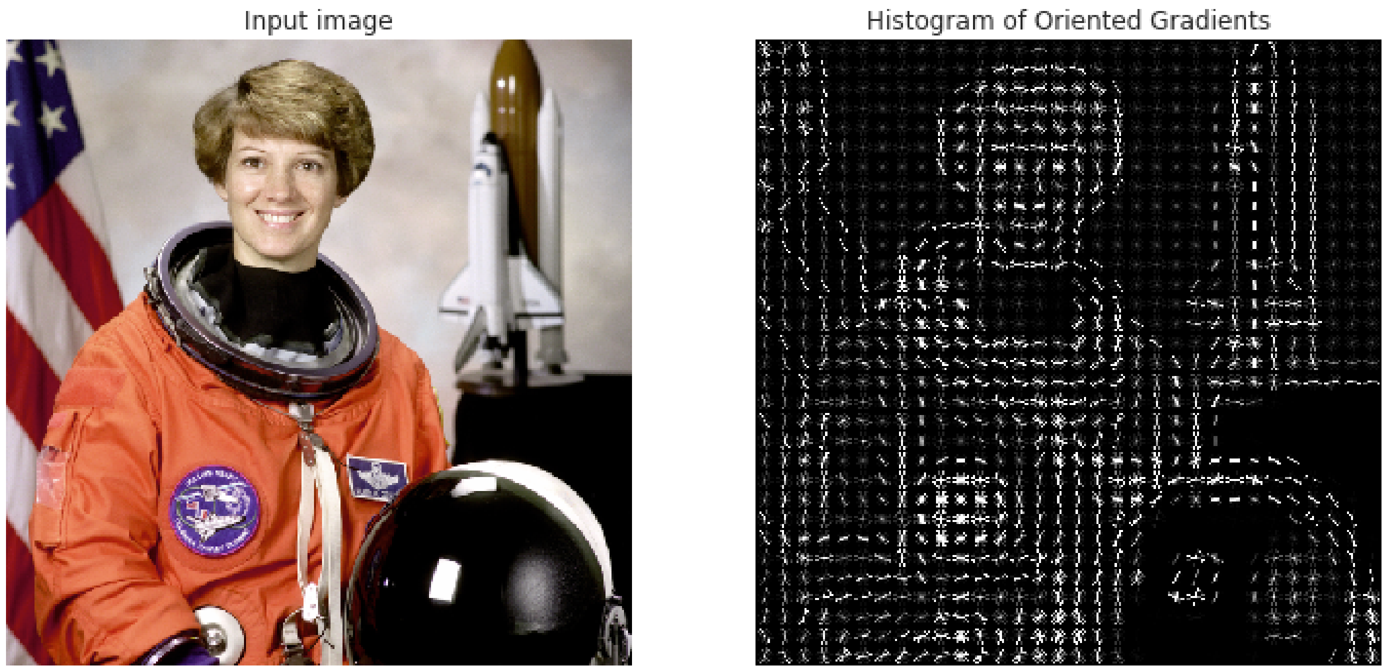


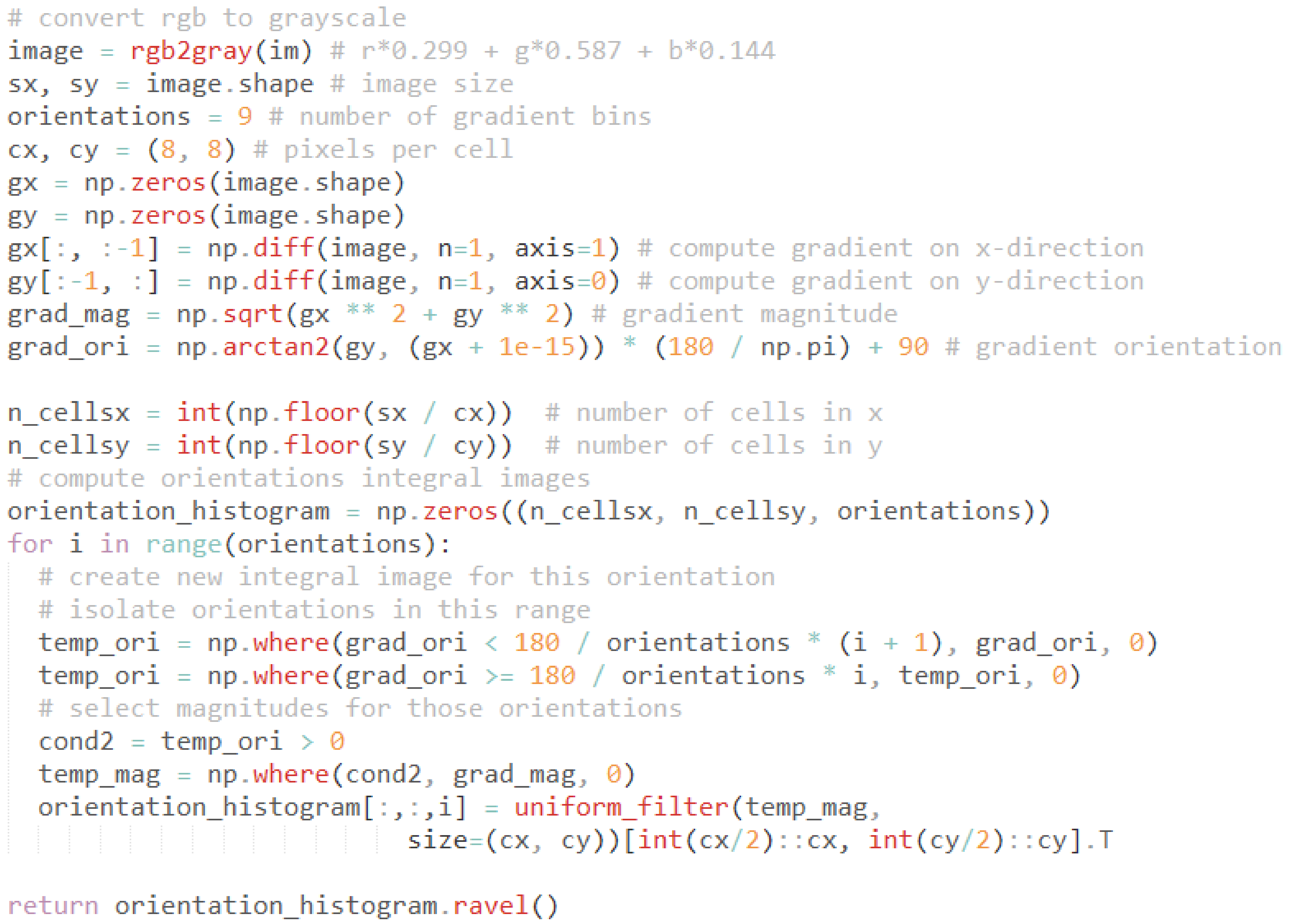


Feature: 1D vector giving the color histogram over the hue of the input image

Histogram of Oriented Gradients

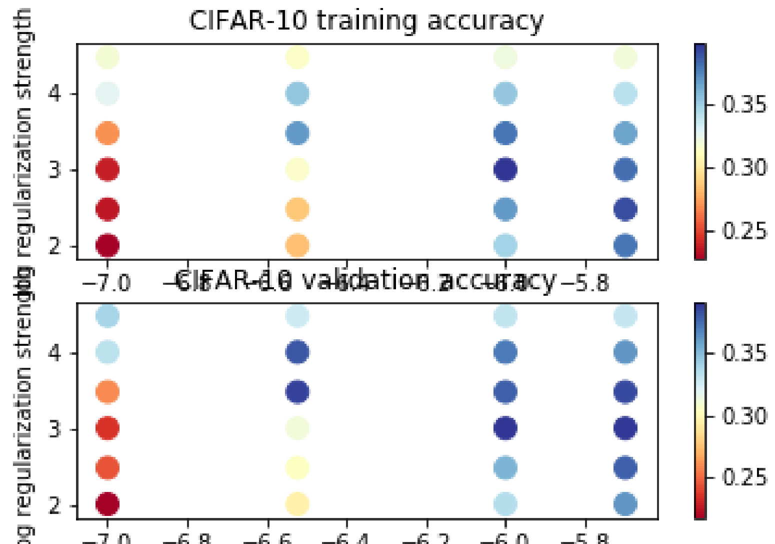


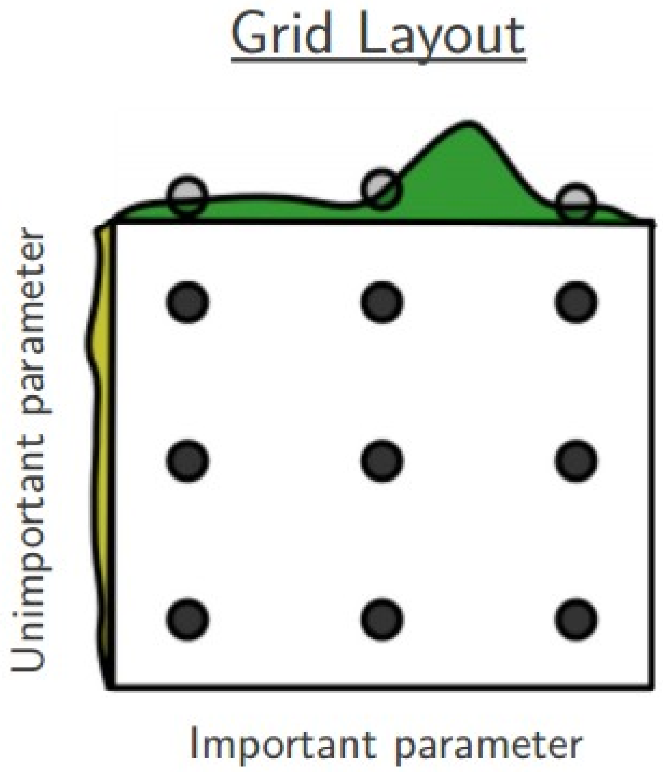


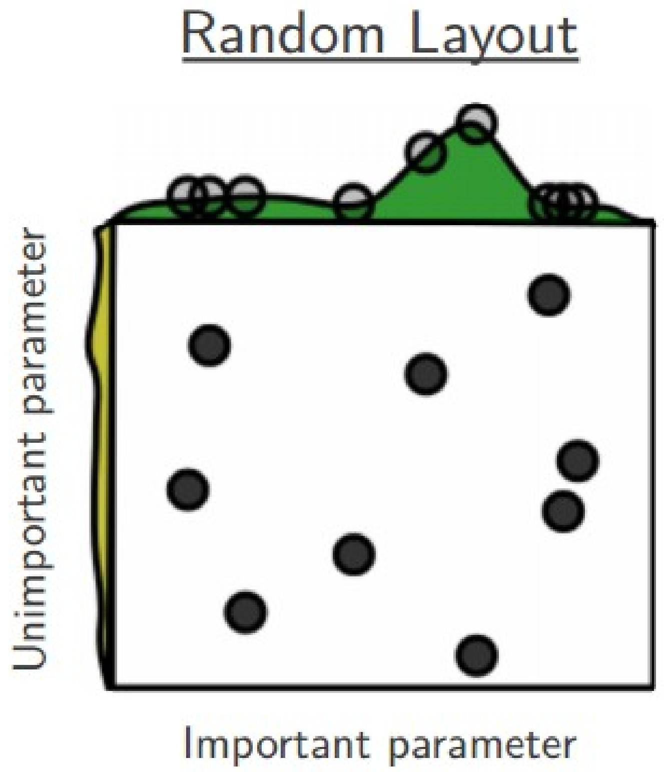


```
# convert rgb to grayscale
image = rgb2gray(im) # r*0.299 + g*0.587 + b*0.144
sx, sy = image.shape # image size
orientations = 9 # number of gradient bins
cx, cy = (8, 8) # pixels per cell
gx = np.zeros(image.shape)
gy = np.zeros(image.shape)
gx[:, :-1] = np.diff(image, n=1, axis=1) # compute gradient on x-direction
gy[:-1, :] = np.diff(image, n=1, axis=0) # compute gradient on y-direction
grad_mag = np.sqrt(gx ** 2 + gy ** 2) # gradient magnitude
grad_ori = np.arctan2(gy, (gx + 1e-15)) * (180 / np.pi) + 90 # gradient orientation

n_cellsx = int(np.floor(sx / cx))  # number of cells in x
n_cellsy = int(np.floor(sy / cy))  # number of cells in y
# compute orientations integral images
orientation_histogram = np.zeros((n_cellsx, n_cellsy, orientations))
for i in range(orientations):
  # create new integral image for this orientation
  # isolate orientations in this range
  temp_ori = np.where(grad_ori < 180 / orientations * (i + 1), grad_ori, 0)
  temp_ori = np.where(grad_ori >= 180 / orientations * i, temp_ori, 0)
  # select magnitudes for those orientations
  cond2 = temp_ori > 0
  temp_mag = np.where(cond2, grad_mag, 0)
  orientation_histogram[:,:,i] = uniform_filter(temp_mag,
                          size=(cx, cy))[int(cx/2)::cx, int(cy/2)::cy].T

return orientation_histogram.ravel()
```

How to choose hyperparameters


CIFAR-10 training accuracy
CIFAR-10 validation accuracy
log regularization strength


Grid Layout
Unimportant parameter
Important parameter


Random Layout
Unimportant parameter
Important parameter