```
import numpy as np
from PIL import Image, ImageFilter
img_filename = 'input_image.jpg'
save_filename = 'output_image.jpg'
########################################
# READ IMAGE FROM FILE
########################################
#load file as pillow Image
img = Image.open(img_filename)
# convert to grayscale
imgray = img.convert(mode='L')
#convert to NumPy array
img_array = np.asarray(imgray)
########################################
# PERFORM HISTOGRAM EQUALIZATION
########################################
##########STEP 1: Normalized cumulative histogram#####
#flatten image array and calculate histogram via binning
histogram_array = np.bincount(img_array.flatten(), minlength=256)
#normalize
num_pixels = np.sum(histogram_array)
histogram_array = histogram_array/num_pixels
#normalized cumulative histogram
chistogram_array = np.cumsum(histogram_array)
#######STEP 2: Pixel mapping lookup table############
transform_map = np.floor(255 * chistogram_array).astype(np.uint8)
#######STEP 3: Transformation#################
# flatten image array into 1D list
img_list = list(img_array.flatten())
# transform pixel values to equalize
eq_img_list = [transform_map[p] for p in img_list]
# reshape and write back into img_array
eq_img_array = np.reshape(np.asarray(eq_img_list), img_array.shape)
########################################
# WRITE EQUALIZED IMAGE TO FILE
########################################
#convert NumPy array to pillow Image and write to file
eq_img = Image.fromarray(eq_img_array, mode='L')
# Detecting Edges on the Image using the argument ImageFilter.FIND_EDGES
image = imgray.filter(ImageFilter.FIND_EDGES)
# Saving the Image Under the name
image.save(save_filename)
```