Human iris center calculation

Eyes are considered to be the most salient and stable feature in the computer vision community. Automatic extraction of eyes plays an important role in many real-world applications such as gaze tracking, face alignment, driver drowsiness detection, face recognition, and human–computer interaction, etc. Several factors that make challenging to detect and track eyes are head poses, lighting conditions, shape and color of eyes, iris movement, imagining quality and conditions, etc. A lot of works have been performed to solve the above problems but still, it remains an open area of research in the computer vision community.

In this task you are going to implement CNN for calculating human iris center. This CNN architecture is proposed in **this paper** as a fully convolutional network which consists of a base network and auxiliary network. It has two losses: reconstruction loss for ensuring the model saves important positional features and second loss is the distance between predicted eye center and ground truth one. The architecture also has skip connection which brings position information to deep layer. Dataset with face images and labels (iris center coordinates + eye corners) is **here**.

You can use cv2 as tool in that task to read, convert color of images, to draw something on image and to do all the other needed operations with images. However you are not restricted to use it and you can use any libraries.

- Preprocess and visualize the dataset:
  - Download dataset. Description of folders and naming are inside dataset folder in README.txt
  - Read all images and convert them to gray with (cv2.ctvColor())
  - Read annotation for images. It contains eye corners and eye centers of 2 eyes for each image.
  - Visualize one image, draw eye corners and iris centers on it
  - Normalize images (divide by 255)
  - Crop eye regions (and resize if needed) to be (48x48) image with the help of eye corners. Do that for all images. It should look like Figure31a)
  - Now data is ready to create a final dataset, which you will use for CNN training. You should create two np arrays X and Y:
    - X contains (48x48) images of eye regions which you crop on previous step
    - Your labels (Y) are coordinates of eye center for each image in X (don't forget to convert iris center from whole image coordinate system to coordinate system of eye region). You should make one more step to cook Y set. For each eye center in Y you should create a (48x48) image (with zero values) and assign value=1 to pixel which coordinate is an iris center. Do it for all images. It should look like Figure1(b)

    Finally, your X and Y sets are lists of 48x48 images. X contains images of eye and Y images with white pixel on the place of iris center.
  - Split dataset

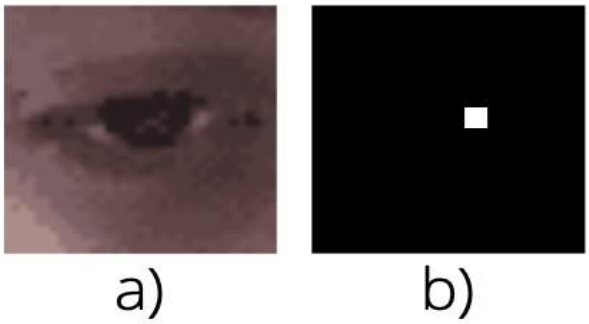

Figure 1

- Build a CNN model using PyTorch.
- Compile and train CNN with different optimizers [sgd, adam, adamax, rmsprop], loss functions [mse, mae] and activations [tanh, relu, sigmoid]. Report best combination.
- Make a prediction for 10 test images. Draw predicted centers on them and visualize it. (You can draw iris center with cv2.circle())

HINTS: Use colab with GPU to speed up training and hyperparameter tuning.