Discussion

My approach for this task is to use blob detection with OpenCV library, which includes the following steps:

- Thresholding based on the min and average brightness values for each image.
- Finding contours and drawing smallest enclosing circles around them. We'll use the center of the enclosing circle to define the coordinates ("location") of each contour.
- Filtering the contours based on their area and perimeter to pick only those contours, which have a size/shape comparable to that of a phone.
- Dealing with special cases where <1 or >1 contour in the image satisfies our constraints.

Using the approach described above, we can detect phone location in 90% of the images. This number can be further improved by a more rigorous DOE to find how different variables and their interactions affect the outcome. The only optimization I've utilized is scanning through the possible space of [min_area, max_area] and [min_perimeter, max_perimeter] values while keeping other variables constant. Other variables were selected through a quick and non-exhaustive trial-and-error process.

Considering the small size of the dataset and not very stringent detection requirements, machine learning with CNNs would likely not be the most efficient way to meet the required detection accuracy.

That said, ML and simpler methods such as Canny edge detection or contour detection are not mutually exclusive. One way to combine them would be to use those simpler methods for image preprocessing and train a CNN on the modified images. Another way would be to use openCV to generate a synthetic dataset containing various features extracted from the image (e.g. parameters of the detected contours), and then train a NN to classify those features as either belonging to an object of interest or not.