To evaluate my text detection system，I will discuss it from 3 dimensions

1. Accuracy
2. Intersection over union
3. Performance metrics

And I will summarize potential improvements

1. Accuracy

For images with clear text on a single background without italics text and artistic fonts, my text detection system maintains a high accuracy rate (90%), i.e., it detects and accurately locates the bounding box containing the full text and accurately identifies the text inside the bounding box.

However, for those images with complex backgrounds (complex background colors and high background noise) or italics and distorted text, my text detection system can only maintain a very low accuracy rate (20%), i.e., most of the bounding boxes containing image text are detected that exceed or do not completely cover the text content, and most of the text content detection results do not match the actual image text content.

For images that contain artistic fonts, my text detection system has an accuracy rate of 0%, i.e., it can not detected at all

1. Intersection over union(IoU)

IoU is a commonly used evaluation metric that measures the degree of overlap between a predicted bounding box and a true bounding box. IoU is calculated by dividing the intersection area by the union area. A high IoU value usually indicates that the predicted bounding box is close to the true bounding box.But it will cost lots of my energy to calculate the precise values,so I will summarize IoU by personal feeling about the output.

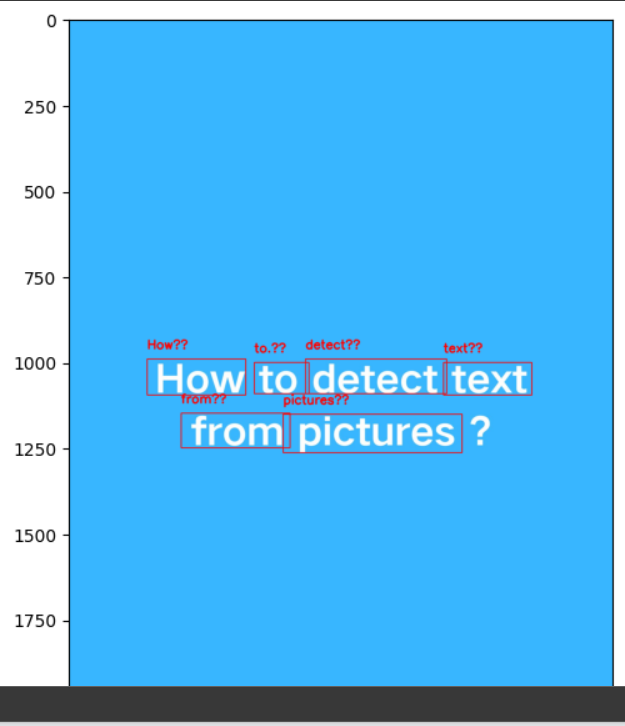
Taking image\_0 as a example:



First, it does not detect the text message "AL-", and secondly, the bounding box detected by the EAST text detector is beyond the bounding box of the actual text in the X axis. Finally, in the Y axis of the detected text bounding boxes of "Miracle-" and "JORDAN" did not contain the complete text information, resulting in the text information of "JORDAN" being detected incorrectly as "IORDAN".

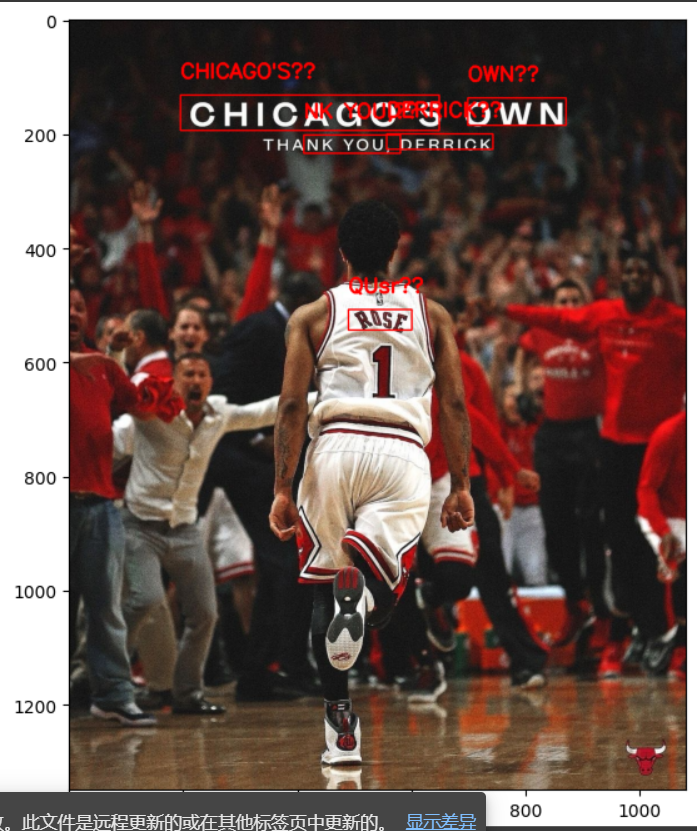
In addition to the above problems, I think the text bounding box detection for "image\_0" is very successful. First of all, it's a image with various noise and having a complex background color, but it detects the slightly transparent text message "Miracle-", and it detects the slightly distorted text message "MONSTER" (the bounding box at the bottom of the clothes).

Taking image\_2 as example:



In this image, the my text detection system successfully detected all the text bounding boxes, and although some of the text boxes overlapped, the text information extracted from the text boxes was not wrong. Overall, my text detection system is very effective in detecting and extracting text information with clear text in a picture in this single background.

Taking image\_3 as example:



This is an image with complex background colors, very high noise, and blurry vista. This image was taken in an NBA arena, and I think it's a very close to the photography subject of everyday life. In this photo, my text detection system detects almost all the text bounding boxes, except for the "THANK YOU" bounding box, my text detection system only detects "NK YOU", and I guess the reason is that the first half of the text has not been detected may be because the background color is too close to the font color. In addition to this, it also detects the bounding box of distorted words on the name tag, which is on the back of the shirt, even though it does not fully cover the text of the name tag on the Y axis.

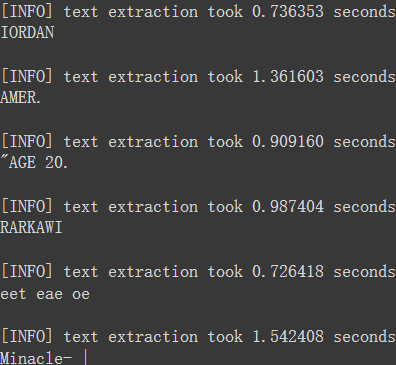
To summarize, the IoU of my detection system is almost 80%.After analyzing the above three cases, I feel that my text detection system can successfully detect most of the text and accurately locate the text bounding box under certain conditions.

1. Performance metric

In this part, I will discuss the processing speed of my text detection system for different types of images to ensure that the system has sufficient efficiency in practical applications, this part is mainly divided into two parts: image text detection speed and text extraction speed from text bounding box.

Taking image\_0 as example:

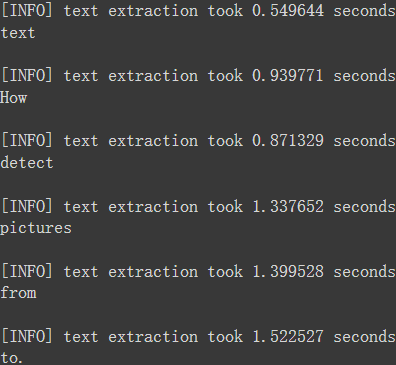




It takes almost 1.5 seconds to detect text, and nearly 1.0 second to extract text from the various bounding boxes.

Taking image\_2 as example:

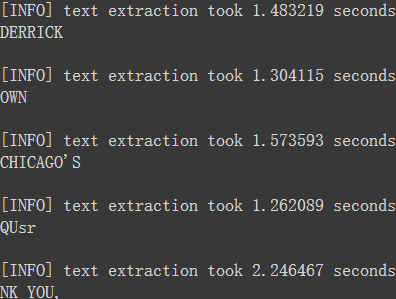




It takes almost 2.7 seconds to detect text, and nearly 1.1 seconds to extract text from the various bounding boxes

Taking image\_3 as example:





It takes almost 3.1 seconds to detect text, and nearly 1.6 seconds to extract text from the various bounding boxes

All in all,After many tests, I found that the text detection time of the text detection system for the above three images will be very different each time, for example, the text detection time of the image\_3 time takes 3.1 seconds, and the next time it becomes 1.4 seconds, probably because the Google Colab platform cannot stably output the same computing power, resulting in the text detection time is very difficult to predict. However, the text extraction time is basically stable at about 1 second to 1.5 seconds.

Potential improvements:

In this Assignment\_1, my text detection system uses EAST as a text detection tool and tesseract as a text extraction tool. I think there are four points of potential improvement for the system

1. **Pre-process works**

There are three methods of image preprocessing that I have learned so far, noise reduction, thresholding and edge detection. For images with various noise and complex background colors, I often use noise reduction methods, such as the Gaussian blurring function, and controlling the value of the "ksize" in the function to make the text in the picture easier to recognize. The thresholding method I typically use for images with a single background color. The boundary detection method is the least effective compared to the above two methods. I believe that if I learned more about image pre-processing, my text detection system will be more perfect.

1. **Text Detection Tool**

This time my text detection system uses the text detection tool is EAST, I know there are also tesseract text detection tool, the above text detection tool compared to EAST, I think its advantage is that the detection speed is fast, because it does not return the possible value of the detected text(scores), so it does not need to filter out the text that is detected wrong, and secondly, it is very suitable for detecting pictures with clear and undistorted text under a single background color, It can clearly locate the text bounding box and extract the text information. The EAST text detection tool does not have high requirements for a series of conditions such as the lighting conditions, image quality and whether the text is distorted, and can accurately detect the text of the picture taken in the natural scene, but its text bounding box positioning cannot completely and accurately cover the text.So I think using different types of text detectors for different pictures is the best option

1. **Text Extraction Tool**

I use tesseract as my text extraction tool, as far as I know, there is also a text extraction tool called easyocr. If you want to improve the text extraction ability of tesseract, I think you should adjust the value of the psm of the config parameter in the image\_to\_string function in the pytesseract library to the appropriate page segmentation mode.EasyOCR returns more values for the likelihood of text detection(confidence value) than tesseract, but the text extraction power is the almost same as tesseract in the three images I tested

1. **Coding Platform**

This time Assignment\_1 compiled using Google Colab. Probably adding a few more CPU cores would improve the detection capabilities of my text detection system(no money to be premium member).