

|  |
| --- |
| **Project Report 2 :**  **Text Image Segmentation for Optimal OCR (Optical character recognition)** |
| **CZ4003 –** **COMPUTER VISION** (Semester 1, AY 2020/2021)  WILSON THURMAN TENG  U1820540H |

# 1. Introduction

## Overview

In this report, various algorithms are implemented to improve the accuracy of optical character recognition (OCR) on 2 images, “sample01.png” and “sample02.png. OCR aims to recognize texts in imaged documents and is one of the earliest computer vision techniques that have been commercialized successfully. An open-source OCR software, [Tesseract](https://github.com/tesseract-ocr/tesseract) is used.

OCR usually involves a series of image processing and recognition tasks including:

1. Text image binarization to convert a colour/grayscale image into a binary image with multiple foreground regions (usually characters).
2. Connected component labelling to detects each binarized character region.
3. Character recognition by using classifiers (E.g. pre-trained neural network).

## 1.2 Dependencies

The following tools will be used in the implementations of this lab report:

* Python v3.6.11 (Source code tested on this version)
* Conda v4.9.2
  + Dependencies listed in ‘src/env.yml’
* Jupyter Notebook
* Tesseract v5.0.0-alpha.20200328

# 1. Metrics

In this project, the following metrics are used:

1. Jaccard Similarity
2. Cosine Similarity
3. Levenshtein Similarity

The 3 metrics are then averaged, and the mean score is used to compare the performance of the pre-processing performed.

## 1.1 Jaccard Similarity

## 1.2 Cosine Similarity

## 1.3 Levenshtein Similarity

# 2. Algorithms

## 2.1 Otsu algorithm

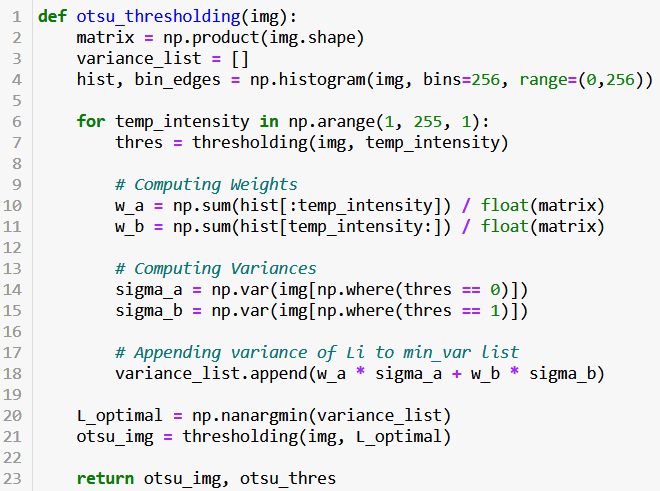
Otsu is used to define a globally optimal threshold under the assumption that the intensity distribution is bimodal. An exhaustive search to minimize intra-class variance is performed on all possible threshold values. The minimization results in the maximization of inter-class variance which allows us to arrive at the desired threshold.



**Li** corresponds to the intensity that is tested.  
**wa** and**wb** corresponds to the probabilities of the two classes separated by a threshold. This is computed as the percentage of pixels in a class.  
**σa** and **σb** corresponds to the standard deviation of the two classes.



**L\*** corresponds to the globally optimal threshold which is computed as the intensity value which minimizes intra-class variance.

**Source code:**  


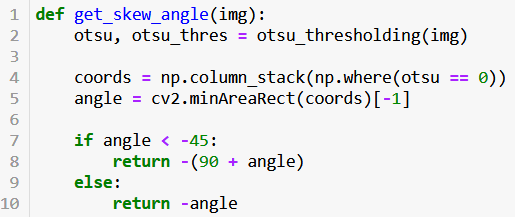
## 2.2 Skew angle algorithm

This algorithm aims to correct the skew angle of the image under the assumption that the text is written in paragraphs which will result in a rectangular bounding box.

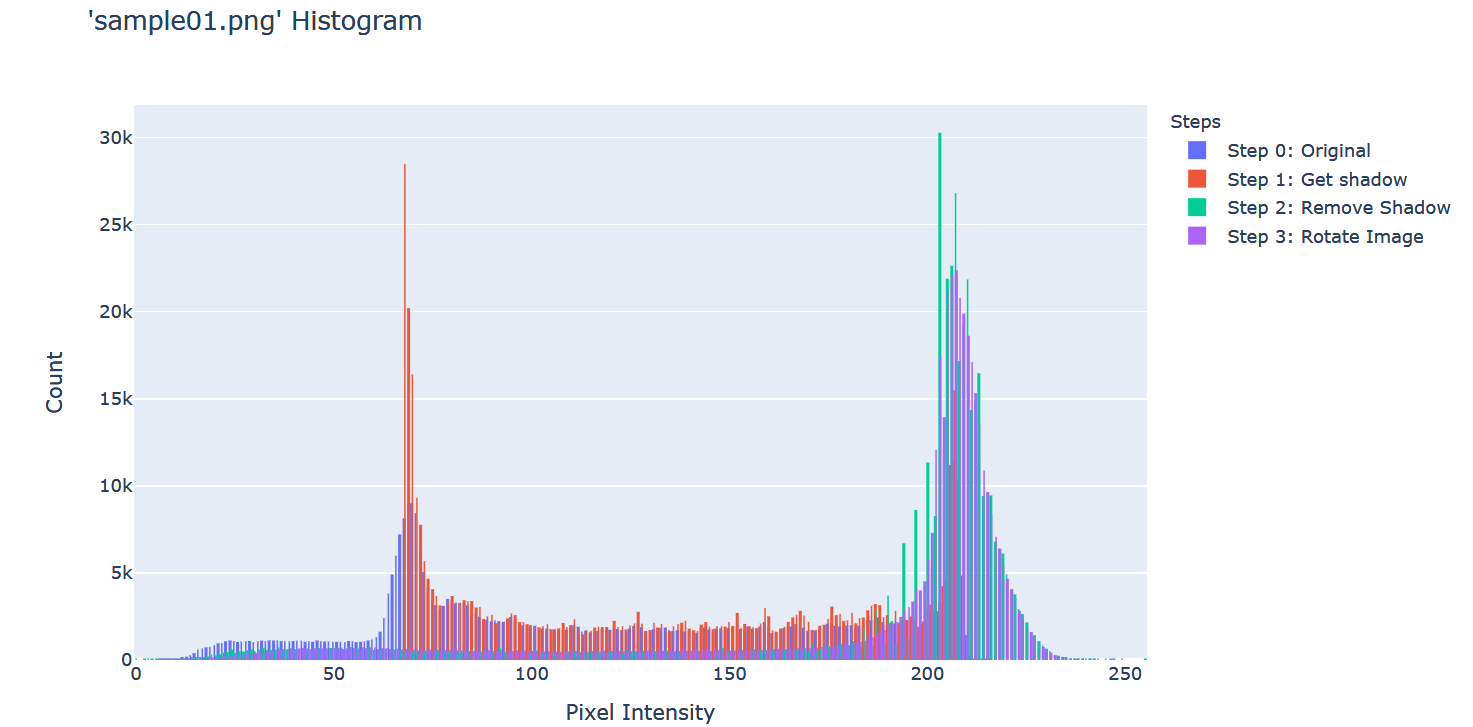
This is done by first performing Otsu thresholding to the image. Coordinates of the pixel of characters (intensity == 0) are then extracted. A minimum area rectangle is computed from these coordinates, and we obtain the skew angle of this computed rectangle.

This method is suitable for images where intensities of the characters and its background are easily separable which allows the computed rectangle to be accurate to the passage.

**Source Code:**

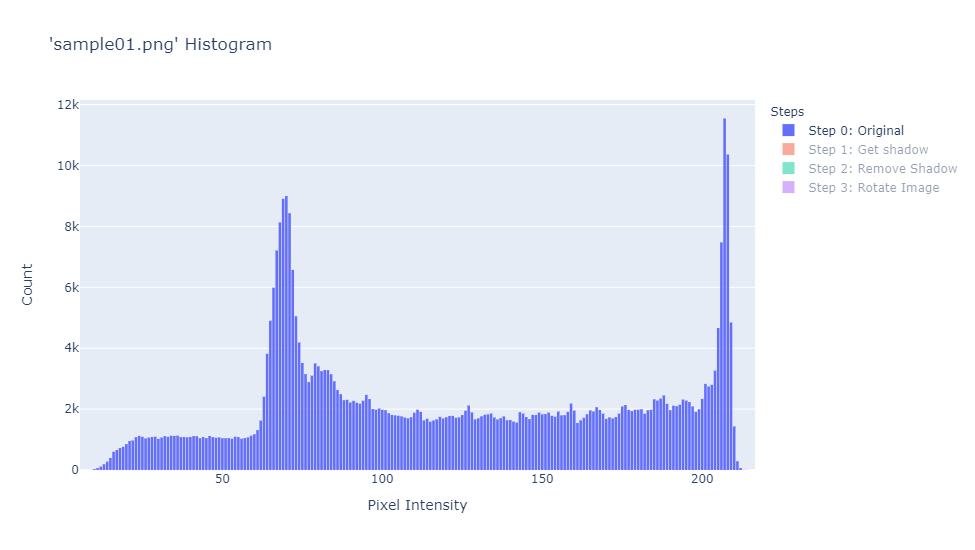


# 3. Pre-processing



The histograms used in this section are from ‘sample01.png’. For an interactive experience with these histogram plots, either ‘OCR\_Project.ipynb’ or ‘OCR\_Project.html’ can be explored.

## 3.1 Step 0: Image Upscaling

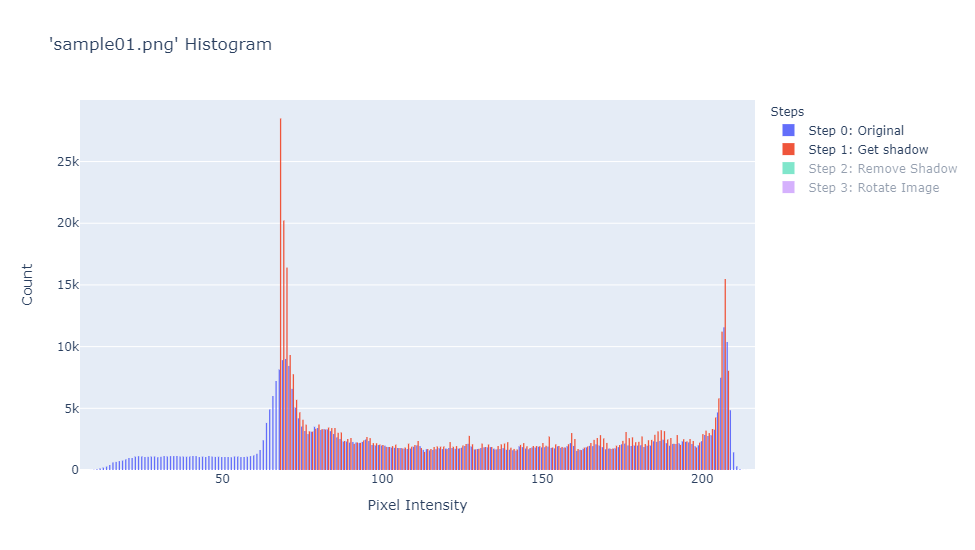


The image is first upscaled between 1.4-1.5x. This is to ensure the image has at least 300 dpi as recommended by Tesseract.

The benefits to upscaling are two-fold:

* This allows for subsequent algorithms to use larger kernels which introduces more flexibility to the shape of kernels that can be used.
* Interpolation between old pixels may create new pixels that fill in gaps between characters.

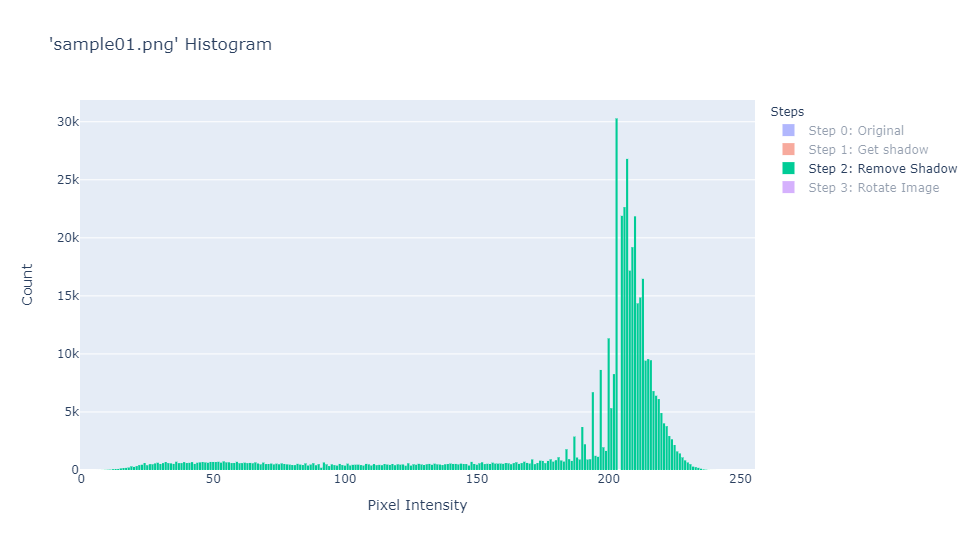
## 3.2 Step 1: Obtain the illumination and shadow of the image



This is achieved by applying Median Blur with a large kernel size to the image. This allows the resultant image to ignore the optical characters and produce the gradient of the image, which explains the illumination and shadow of the image well.

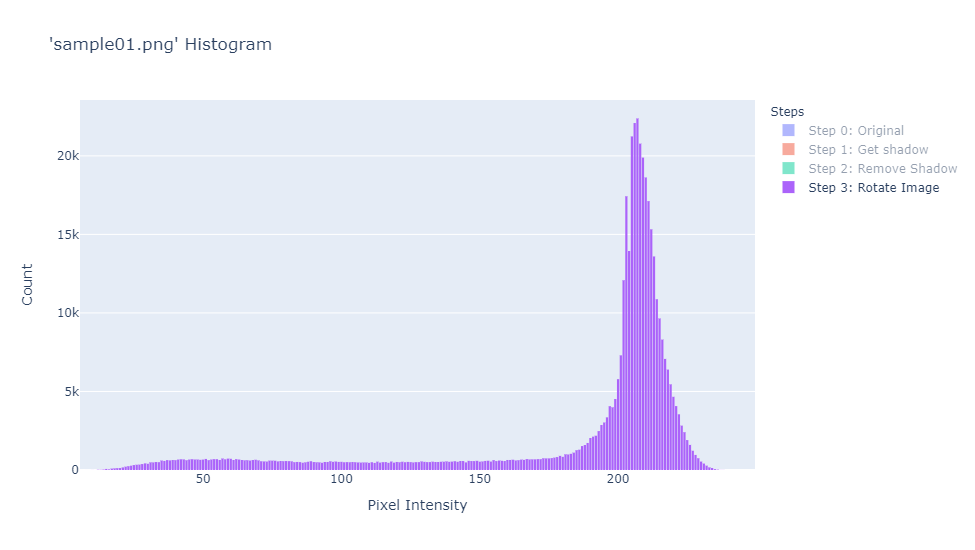
It can be observed from the histogram plot above that the shadow and the original image overlap for many intensities, hence we can conclude that a global threshold technique would be insufficient to identify an optimal threshold.

## 3.3 Step2: Remove illumination/shadow from image



This step reduces the effect of illumination and shadows. The original image is divided by the shadow obtained in the previous step and contrast stretched to the original [0,255] range. This darkens the previously illuminated parts of the image, while the dark parts of the image remain at similar intensities.

## 3.4 Step 3: Rotate image (if required)



Finally, in the last stage of pre-processing, we utilize the skew angle algorithm explained in section 2.2 and apply an affine transformation to our original image.

It is interesting to note that the histogram of the rotated image follows a much smoother curve as compared to before rotation.

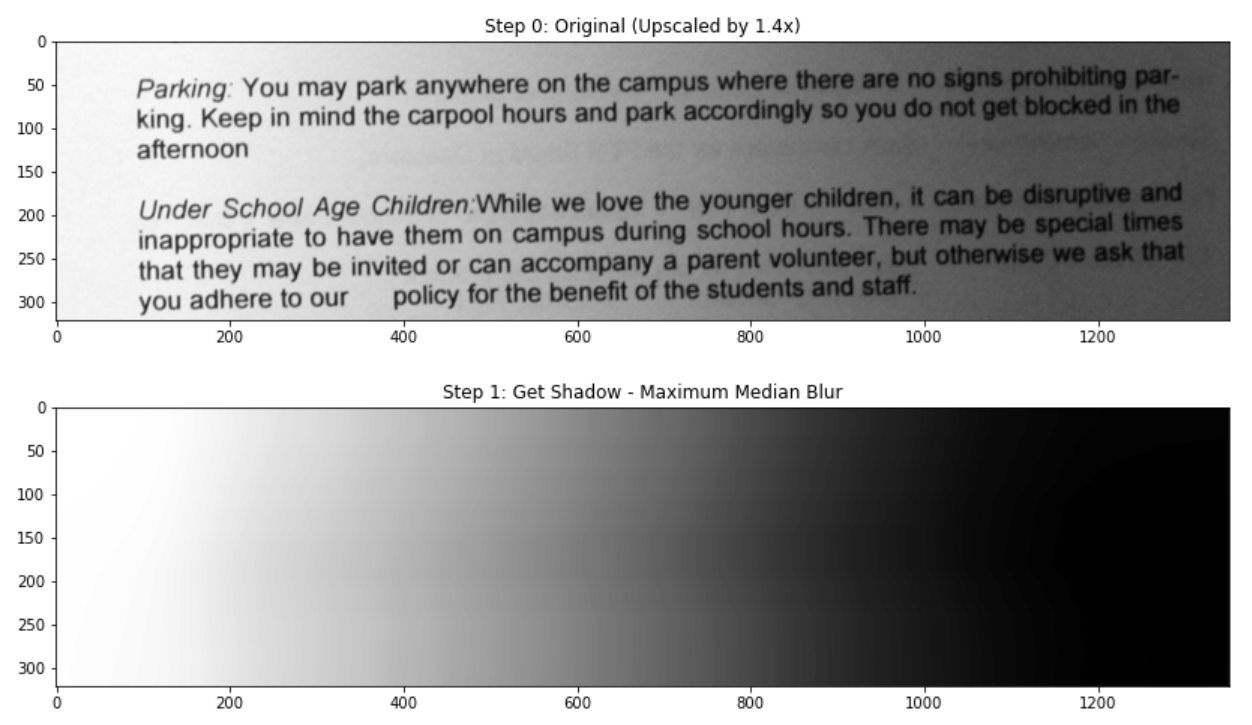
## 3.4 Conclusion

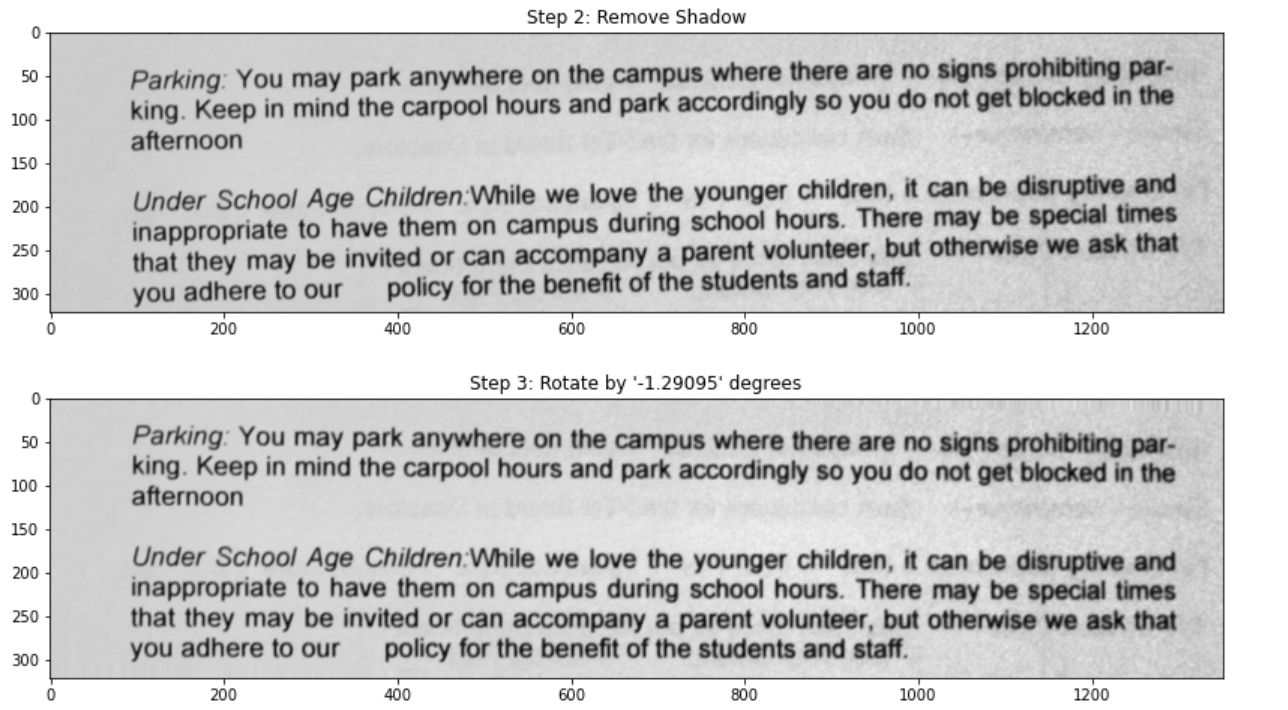
It can be observed in that the final histogram obtained is not bimodal, therefore binarizing the image may not be useful at all for obtaining accurate OCR results.

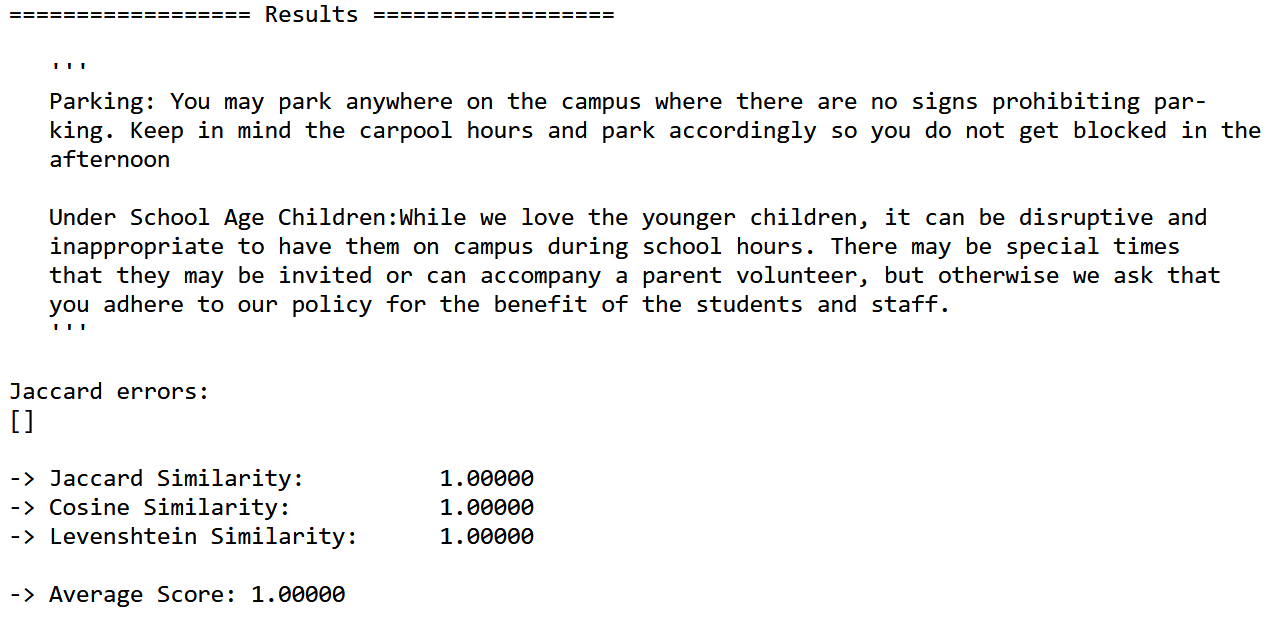
# 4. Results

## 4.1 ‘sample01.png’ results (Best score achieved – 100%)

The following show the results of the pre-processing steps outlined in the previous section as well as the score achieved.

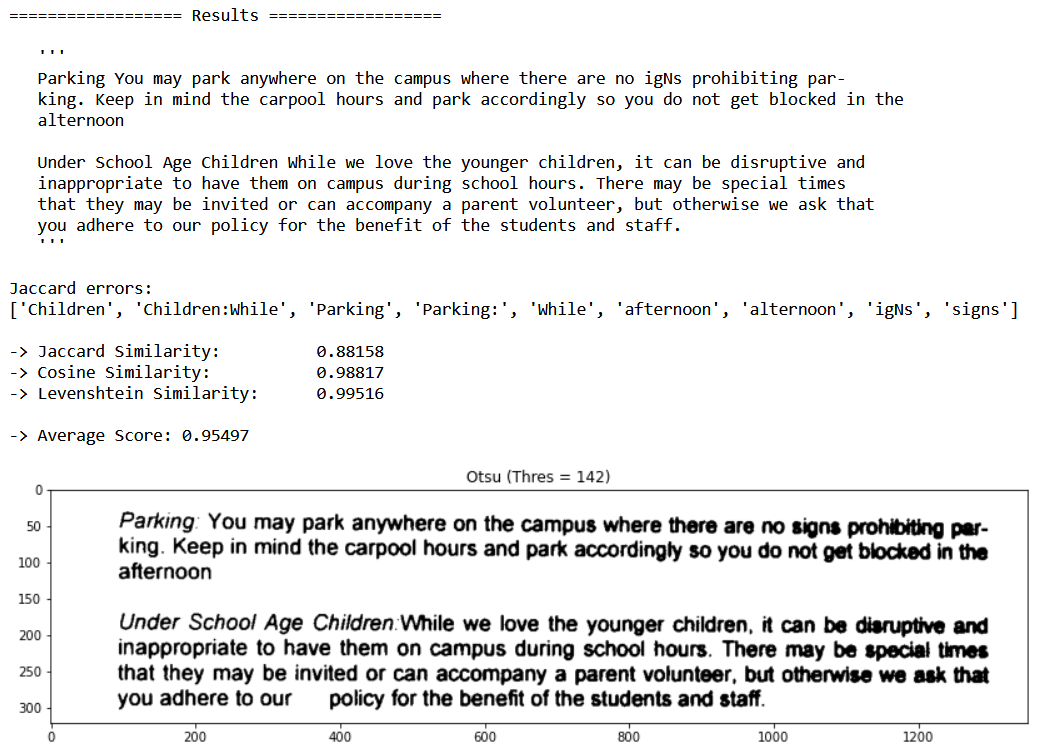






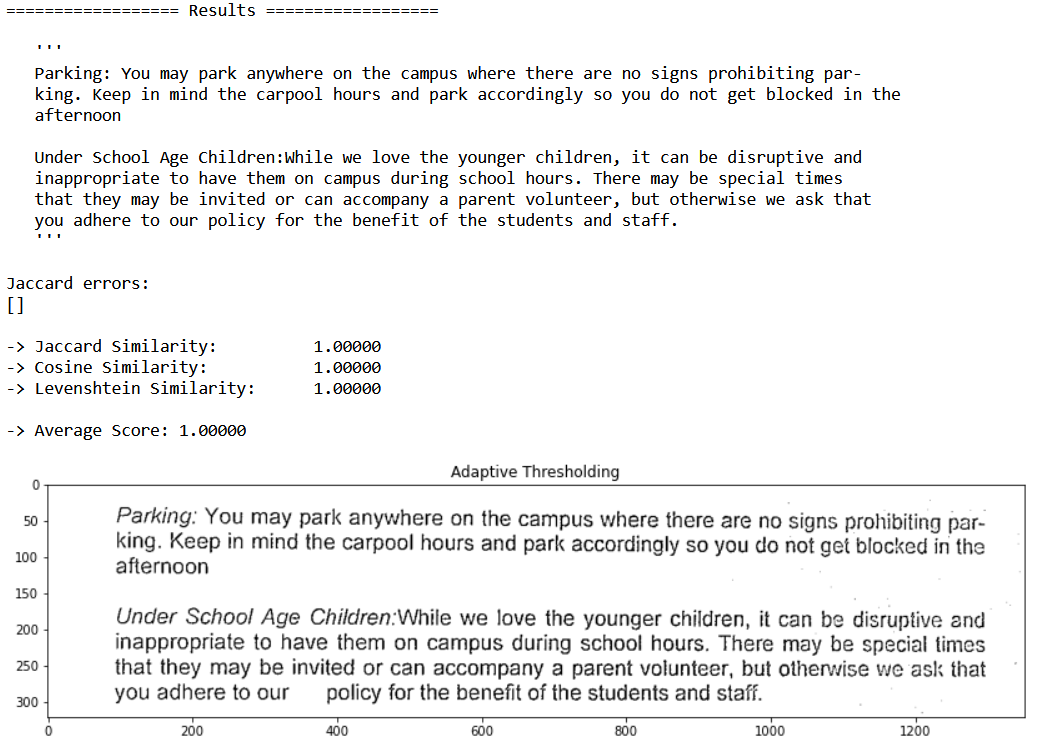
Experimentation with Otsu and adaptive thresholding on the processed image was also attempted but these binarization algorithms did not improve the score. This could be attributed to the unimodal distribution of the processed image observed.

### 4.1.1 Otsu Thresholding



It can be observed that Otsu thresholding results in thick strokes caused by the remaining slight skew which affects the accuracy of the OCR.

### 4.1.2 Adaptive Thresholding



Adaptive thresholding is also applied to the processed image. This allows for locally optimal thresholding which results in a clean binarized image.

## 4.2 ‘sample02.png’ results (Best score achieved – 99.04%)

|  |  |
| --- | --- |
|  |  |
|  |  |

# 5. Discussion (Optional)

This section discusses techniques to improve on OCR.

## 5.1 Types of image degradation