Food Item Image Display Using Food Menu Image File

27.11.2018

—

Team - RGB
Aniket Bansal - 20161152
Rohan Shrivastava - 201431227

Abstract

Optical character recognition is the process of extracting text or character from an input image file using various methods of digital image processing.

It is very useful as it compensates for manual detection of texts from written or printed hard documents. It is also useful in handwriting detection of multilingual documents. A general OCR application takes the input image and gives out detected characters. Its uses extends to IOT where effective OCR can detect any road sign and alert driver on a road.

Input Image -> OCR -> Output Text

But using inbuilt OCR function directly without classifying text region in an *unstructured* image like as food menu will give faulty results.

In this project we have classified text and non-text regions, then removing the false detection progressively and then predicting the text in the detected regions. Finally using google custom search api to output first image found from the search result of food item.

Introduction

Problem Statement

Given a picture of a restaurant menu (table version or one displayed on a wall), show corresponding food item pictures.

Motivation

Optical Character Recognition in food menu is a challenging task in computer vision. This is because food menu image has too much variability in font, graphical variations, more non-text and less text region. The main objective is to train a classifier for text region to give accurate and complete results.

Also finding the right corresponding food image is a difficult task on google so we tried applying different tags with the search.

Overview

Basic steps involved are-

1. Train a classifier for text vs non-text region.
2. Recognize true character candidates and remove all false positive (Non text) from our detected candidates from previous step.
3. Detect accurate text in the discovered text regions.
4. Classify text into food item text and non-food item text.
5. Use google custom search api to output desirable food items corresponding to text.

Procedure

1. ## Detect Candidate Text Regions Using MSER

   The MSER feature detector works well for finding text regions. It works well for text because the consistent color and high contrast of text leads to stable intensity profiles. Notice that there are many non-text regions detected alongside the text.

2. ## Remove Non-Text Regions Based On Basic Geometric Properties

   Although the MSER algorithm picks out most of the text, it also detects many other stable regions in the image that are not text. You can use a rule-based approach to remove non-text regions. For example, geometric properties of text can be used to filter out non-text regions using simple thresholds. Alternatively, you can use a machine learning approach to train a text vs. non-text classifier. Typically, a combination of the two approaches produces better results.

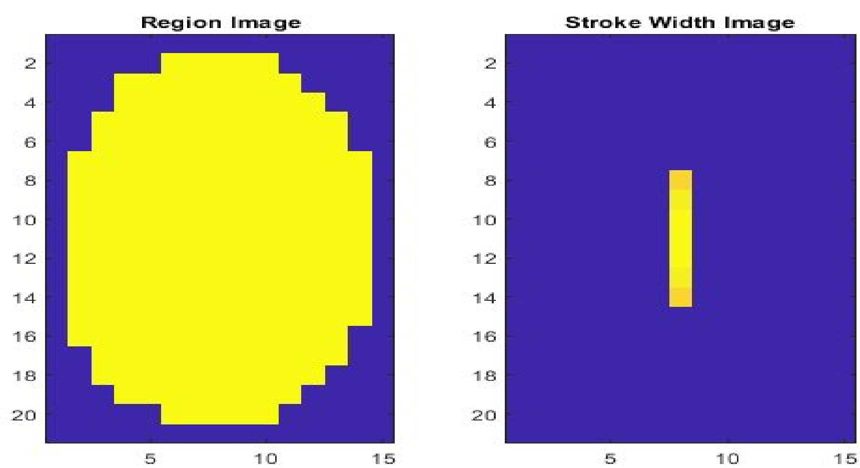
   There are several geometric properties that are good for discriminating between text and non-text regions , including:

   - Aspect ratio
   - Eccentricity
   - Euler number
   - Extent
   - Solidity

3. Remove Non-Text Regions Based On Stroke WIdth Variation

Another common metric used to discriminate between text and non-text is stroke width. Stroke width is a measure of the width of the curves and lines that make up a character. Text regions tend to have little stroke width variation, whereas non-text regions tend to have larger variations.

To help understand how the stroke width can be used to remove non-text regions, estimate the stroke width of one of the detected MSER regions. You can do this by using a distance transform and binary thinning operation.



In the images shown above, notice how the stroke width image has very little variation over most of the region. This indicates that the region is more likely to be a text region because the lines and curves that make up the region all have similar widths, which is a common characteristic of human readable text.

4. Merge Text Regions For Final Detection Result

At this point, all the detection results are composed of individual text characters. To use these results for recognition tasks, such as OCR, the individual text characters must be merged into words or text lines. This enables recognition of the actual words in an image, which carry more meaningful information than just the individual characters.

One approach for merging individual text regions into words or text lines is to first find neighboring text regions and then form a bounding box around these regions. To find neighboring regions, expand the bounding boxes computed earlier with regionprops. This makes the bounding boxes of neighboring text regions overlap such that text regions that are part of the same word or text line form a chain of overlapping bounding boxes.

Now, the overlapping bounding boxes can be merged together to form a single bounding box around individual words or text lines. To do this, compute the overlap ratio between all bounding box pairs. This quantifies the distance between all pairs of text regions so that it is possible to find groups of neighboring text regions by looking for non-zero overlap ratios. Once the pair-wise overlap ratios are computed, use a graph to find all the text regions "connected" by a non-zero overlap ratio.

Finally, before showing the final detection results, suppress false text detections by removing bounding boxes made up of just one text region. This removes isolated regions that are unlikely to be actual text given that text is usually found in groups (words and sentences).

5. Recognize Detected Text Using OCR

   After detecting the text regions, use the ocr function to recognize the text within each bounding box. Hence we have found a list of food item with their description which can be googled to find corresponding food image. Note that without first finding the text regions, the output of the ocr function would be considerably more noisy.

6. Use Google Custom Search API To Display Corresponding Food Item Image

   Google provides us Google Custom Search API to integrate his search engine into our apps. What we need is an API key and a Search Engine Id. We can use Google API client for this purpose.

   Use different searching methods on the google by applying a common tag in the search and test it.

Functioning Examples-

I. Food Menu Image

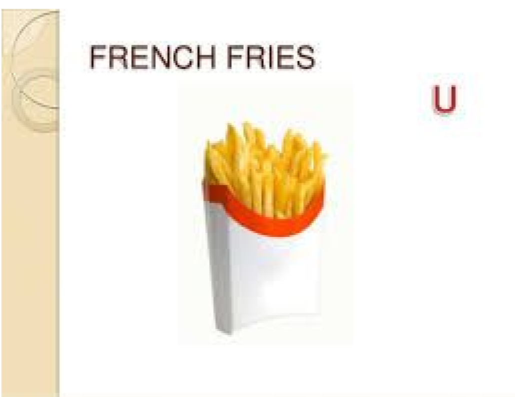
Input-

Output-


FRENCH FRIES
U


Peanuts and Peanut Oil Card
I have a life-threatening allergy to peanuts and peanut oil.
If I eat this food or any food that has been cooked with it or touched it, I will need immediate medical attention.
Does this food contain peanuts or peanut oil?
SelectWisely


SODA
POP


shutterstock.com · 200523716


Coca-Cola

Result Of the Project

We implemented the designed algorithm to classify text and non-text region, reduced false detection regions, applied ocr to find text, classified food item names by region thresholding, reduced false food names and finally searched google with effective searching methods.

Observation and Constraints of the Project

By testing the code for different food menu images we observe that there are certain limitations which has to be overcome.

Example - Even after detecting the text most accurately, there is a chance that the output image result is faulty or not well informative about the food. This happened mainly due to lack of information present in the name of the food. Therefore menu image with explanation of the food name gives more accurate result than with less information. Also faulty results are observed even after applying effective search methods.