Computer Vision Word Search Solver

By Josh Keegan
Supervised by Dr. Jon Barker

Abstract

This project aims to create a Computer Vision system capable of detecting word searches and then using Artificial Intelligence techniques to solve them. There is no one correct process to solve this problem, making the project very experimental in nature.

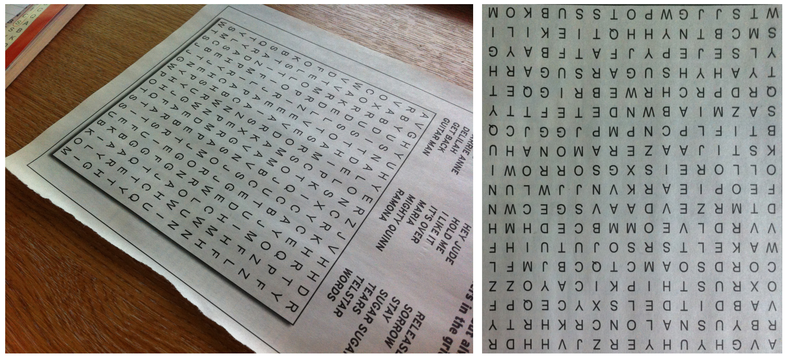
The system manages to detect and solve 88% of word searches that are surrounded by a bounding box in order to aid detection. Word search detection has been identified as the main area for further work; if a perfect word search detector were to be devised, the overall system performance would rise to 96% of word searches being solved correctly. There are no other computer vision systems detecting and solving word searches known to the author at this time for these results to be compared with.

Word Search Detection

- Aims to find the word search grid in an input image; see Figure 1(a)
- Locates the word search using its rectangular bounding box
- Returns a Bitmap image of the word search found; see Figure 1(b)

Basic Process:

1. Greyscale and binarise the input image (using Bradley Local Thresholding)
2. Perform blob recognition, find blobs that are roughly quadrilateral and meet a minimum width and height
3. Extract a Bitmap of each quadrilateral, correcting for skew
4. Score each candidate Bitmap using a model of a word search (uses the Segmentation stage and checks the standard deviation of row height, column widths, gaps between rows and gaps between columns)
5. Returns the Bitmap that scores highest



(a) Input Image (b) Output Image

Figure 1: Example input and output of the Word search Detection stage

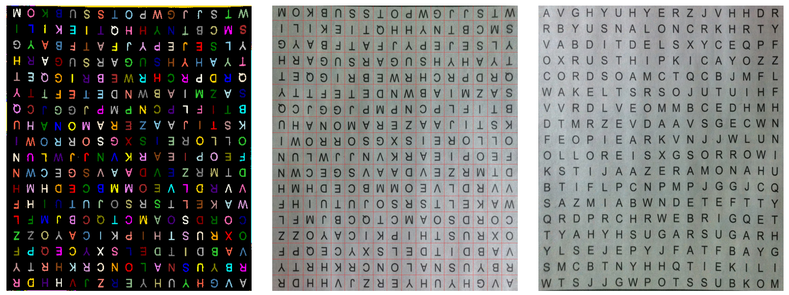
Segmentation

Segmentation locates each row and column in a word search. Several different algorithms were implemented for this stage; this problem became the main focus of the project.
The algorithm that performs best for this stage uses blob recognition with a minimum and maximum blob height, followed by removing blobs considerably smaller than the mean blob width and height. This gives the size and location of each character in the word search; see Figure 2 (a). Then the position of each column is determined by grouping the characters where they overlap in the X direction and the same is done in the Y direction to find the rows. Figure 2 (b) shows a word search having been segmented by this method.

Rotation Correction

Images of word searches extracted by the Word search Detection stage can be in any of four possible orientations: the top of each character can face the top, right, bottom or left of the image; see Figure 1(b) for a sample Word search Detection output. Given any of these inputs, Rotation Correction will return the same image, but rotated such that the top of each character faces the top of the image.

It works by classifying each character in the image in each of the four possible orientations and for each orientation summing the highest class score for each character. The image in the correct orientation is the one that scores highest, as in each of the other orientations the classifier input will have not looked like any character.



(a) Character blobs (b) Segmented (c) Rotation Corrected

Figure 2: Word search Image being segmented and having its rotation corrected

Character Image Extraction, Feature Extraction and Classification

Character Image Extraction, Feature Extraction and Classification are three stages that are always used in combination with one another. Together they take an image containing a single character and return a score for each possible class (the characters A-Z).

Character Image Extraction identifies the region of the image that actually contains the character, removing whitespace and noise.

Feature Extraction measures some features of the image, aiming to quantify things about the image that will aid the classifier in determining its class. The best Feature Extractor for this project uses Principal Component Analysis.

Classification takes the features and uses them to produce a score for each possible class. Classification is done by a single-layer Neural Network using a Bipolar Sigmoid activation function and trained with Backpropagation learning.

Word Search Solver

The Word search Solver takes the character class scores from the Classification stage, along with a list of words to be found and uses them to solve the word search.

For each word to be found, each grid position is tried as a potential starting point. The word is tried in each of the eight possible directions and a score for each placement is calculated by summing the class scores of each grid position used, for the character it is used as.

Once all of the possible positions for each word have been scored, the solution is found by finding the overall solution that gives the highest score, placing each word exactly once. However, the solution must not use the same position in the grid as more than one character; this makes the problem intractable. However, seeing as each word should be in the word search only once, the correct solution for each word should score as one of the highest. The best solution is found by ordering the candidate positions for each word by their scores descending and then performing a breadth-first state space search of the possible solutions, with the starting node being a solution made up of the position that scores highest for each word.

In order to prevent the entire tree being searched for a solution (or a very large chunk of it) when the class scores are “garbage” (caused when an earlier stage has failed, e.g. the word search not being found correctly), there is a maximum search depth specified. If all nodes to this maximum depth have been evaluated and no valid solution (one without collisions) found, then the solution returned will be the top scoring position for each word, even though this has collisions. This is done for evaluation purposes and would not be desirable in a production system.

Results

Several modules and combinations of modules were evaluated in order to give a clear picture of the strengths and weaknesses of the system.

Word Search Detection

All different models of a word search (segmentation algorithms) were evaluated. The best model found the word search in **88%** of images.

Word Search Segmentation

Each of the seven different segmentation algorithms implemented were evaluated both with and without having erroneously small rows and columns removed from their results (ones that were significantly smaller than the mean). Three algorithms managed to calculate the number of rows and columns correctly **100%** of the time when erroneously small rows and columns were removed.

However, a large increase in performance between the standard algorithm and after erroneously small rows and columns have been removed indicates oversegmentation problems that may lead to the actual pixel indices of each row or column being out of position, potentially by enough to not entirely contain the characters. One algorithm (using blob recognition) did not show this, suggesting genuinely high performance by managing to get the segmentation correct 99% of the time without having to have any erroneously small rows or columns removed.

Feature Extraction and Classification

There was only one sort of classifier used in this project, but it had to be trained separately using the data from each different feature extraction technique. Using all of the features from PCA feature extraction performed the best, with **98.26%** of character images being classified correctly.

Rotation Correction

When evaluated independently rotation correction returned the image in the correct orientation **100%** of the time. This module is highly robust to single character misclassifications.

Full System

The full system was evaluated from start to finish. Where a single algorithm had performed better than others in earlier evaluations it was fed-forward for that stage into the final system. Overall, the final system manages to return the correct solution for **88%** of images.

Post-Detection Stages

The score for the full system and the score for the word search detection stage on its own are very similar, with other stages of the system performing significantly better. This gives the impression that the word search detection stage is the main weakness in the system.

In order to test whether the word search detection system was the main source of error, the rest of the system was evaluated by passing the correct result for the word search detection stage through the remaining stages. In this evaluation, the system found the correct solution to 96% of the word searches in the evaluation data. This confirms that the word search detection stage is the main source of error, but also shows that other areas of the system are not perfect.

Further Work

The performance of specific aspects of the system is good, and the overall system performance reflects this. However, the problem of word search detection has been identified as an area for further work, with that one stage accounting for the majority of errors in the system. In addition to improving the performance of the current detection system, further work can be done here in order to remove the requirement for word searches to be bounded by a rectangle.

This project did not attempt to address the problem of finding the words to be found due to there being no standard place for them on the page. Sometimes words are in different fonts, on different pages and can even be highlighted in poems or given as clues. This therefore is also potential further work, although it would be a very ambitious addition to the project.