Jack Cheng Ding Han 150159519 ACA15JCH

Introduction

My wordsearch function begins by pre-processing the test data. The get_characters function I've written takes in a square array and returns an array that can be inputted into the classify function. For test1 or test2, it takes a 450 by 450 array and turns it into a 225 array of 30 by 30 square arrays, ordered row by row. Each 30 by 30 subarray can then be turned into a 900 element feature vector. This is always the first step.

If the reduced argument is set to True, then training_dat variable to be inputted into the classifier as the training data will be the train_dat argument reduced to 10 features by default it is the first 10 features selected by PCA however, there is a sixth parameter for a list of indices for specific features selected by PCA.

The classify function is called with training_dat as the training data and the testing data as preprocessed or preprocessed reduced to 10 features. The classifier outputs a 15 by 15 array of the labels which is stored in classified_mat.

It proceeds to find every word in the words_to_find parameter. The search for a word on classified_mat is in three stages; horizontal, vertical and diagonal. Horizontal and vertical searching is a case of looping through fifteen rows or columns, respectively and calling search on the array corresponding to the row or column each iteration. Before searching an array of indices for test1 or test2 because the search function used later will take diagonals and such that will use different indices for the same labels. The function builds a list of tuples that store the likeliest place a line could begin for a particular word. Explained further in the next paragraph.

The search function goes along the array with the same number of steps as the length of the word and stores the number of characters matching as parts of a tuple. The first part is the original index of the label in test1 or test2, the second part of the tuple is the number representing the label in classified_mat and the third part is how many labels following it in the array currently being search match. If the number of labels matched, is equal to the length of the word, then it is an exact match and is returned immediately with a dummy value (0,0,0) to signify an exact match. Otherwise, it adds it to a list of tuples and returns them.

The trav_diag function searches through classified_match including diagonals parallel to the leading diagonal and perpendicular to the leading diagonal using the previously mentioned search function. It begins at index 0 which is the leading diagonal calculates tuples where the third part is the number of labels matching. Once it has finished a diagonal it moves in the positive direction to the next diagonal but it stops if the next diagonal is shorter than the word and begins searching in the negative direction and when a diagonal shorter than the word is encountered again, there is a recursive call to do the whole process again on diagonals perpendicular to the leading diagonal. Again, if exact matches are found, they are returned with a dummy value.

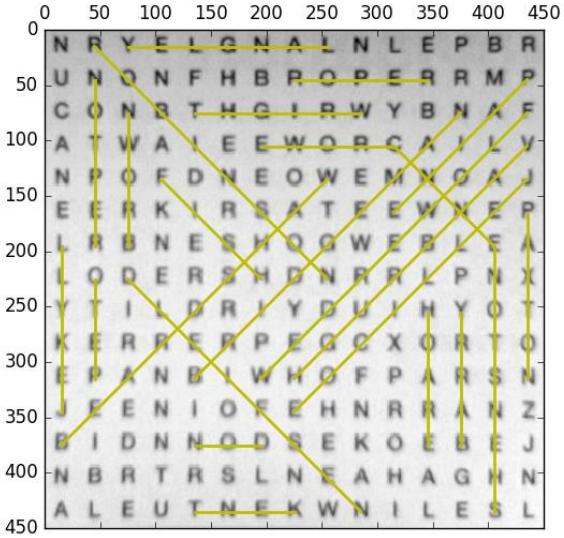
If no exact matches were found after the three stages of searching, the function finds the likeliest indices to draw a line by sorting the list of tuples built up through the three stages of searching, selecting the biggest one. For exact matches the tuple at index 1 in the list is selected as the tuple at index 0 would be the dummy value.

Jack Cheng Ding Han 150159519 ACA15JCH

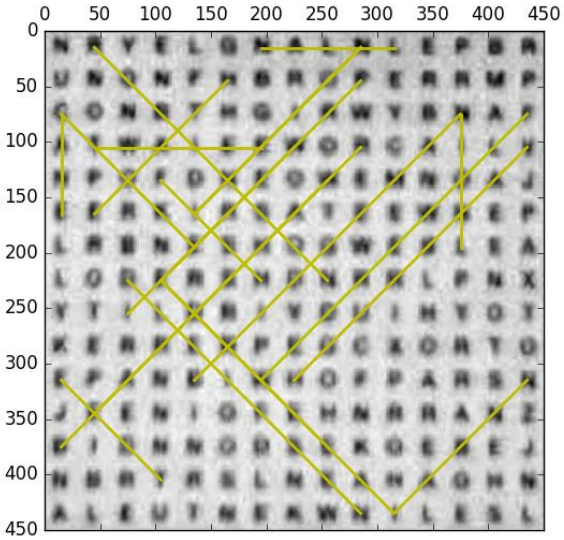
Originally, my reduce function would select the first 10 features (0≤i≤9) from PCA on training data. The current code however, adds 2 to reduce_by to calculate the first 12 features (0≤i≤11) then, using nested list comprehensions, selects 10 features, each at the following indices: [1, 2, 3, 5, 6, 7, 8, 9, 10, 11] wordsearch found the same number of words if 14 replaced 8. However, the classify function performed poorer, only getting 122 labels correct. From observations, sets of features that didn't use any of first 15 features (0≤i≤14) from PCA had bad results for test2, well below 50% of words. Using itertools, I looped a stripped down version of wordsearch (i.e. I commented out all the calls to print, plt, drawline etc. and made it return a tuple consisting of the number of labels correct and number of words correct) to test all possible subsets of the first 15 features that are of length 10 to find the best one, and that is how I got the indices used in the final version. Below are comparisons involving wordsearch. The obsolete results can be created again by leaving out the sixth argument when calling wordsearch. The classifier with reduction also got 94% using train1_data and test1_data.

First 10 Features from PCA

Solving test1 With Reduction
217 letters were correctly labelled which is about 96% of the letters.
23 out of 24 found correctly.

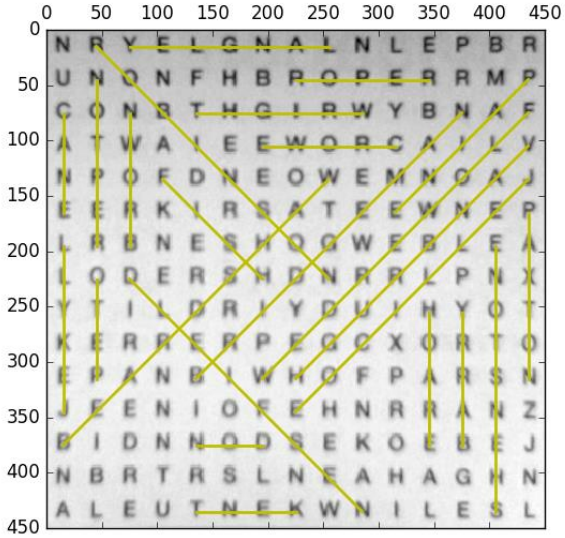


Solving test2 With Reduction
81 letters were correctly labelled which is about 36% of the letters.
8 out of 24 found correctly.

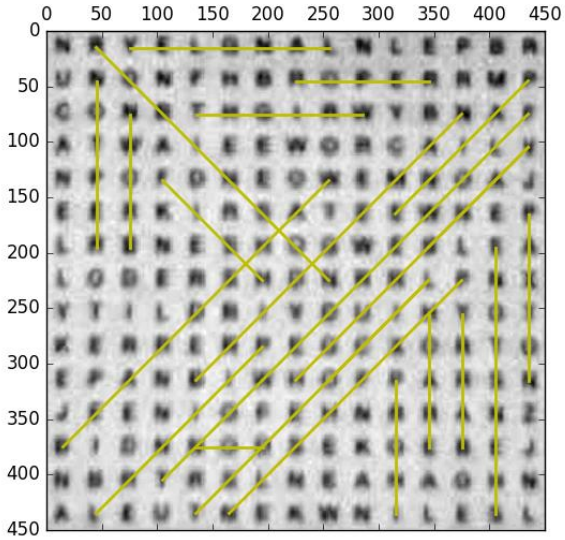


First 12 Features from PCA Except 0 and 4

Solving test1 With Reduction
224 letters were correctly labelled which is about 99% of the letters.
24 out of 24 found correctly.



Solving test2 With Reduction
141 letters were correctly labelled which is about 62% of the letters.
17 out of 24 found correctly.



Jack Cheng Ding Han 150159519 ACA15JCH

Trials

Trial 1

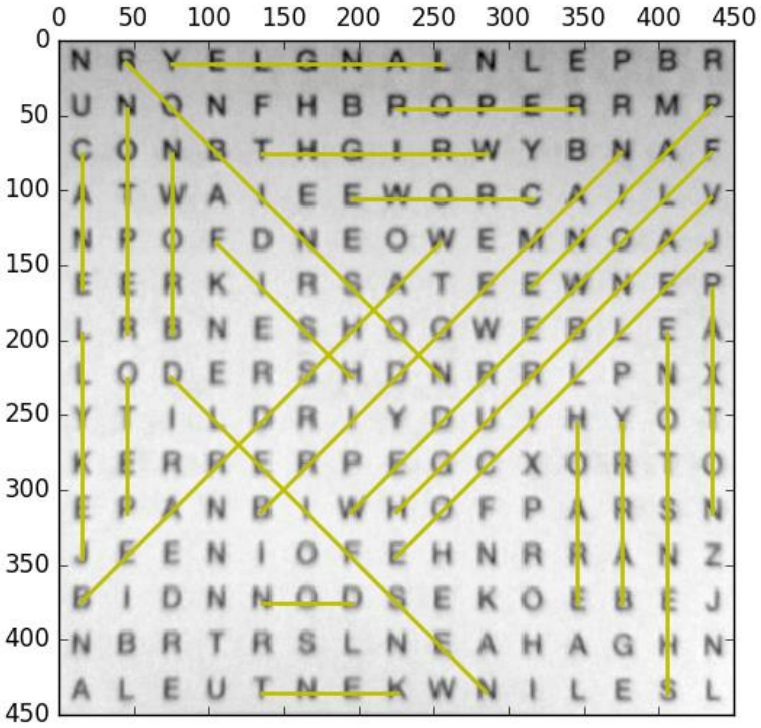
Without dimensionality reduction, the classifier only classified 1 letter incorrectly. The wordsearch function, being able to find the best matches for words that aren't exact matches, manages to draw every line on the image correctly.

Input:

```
wordsearch(test1, words, train_data, train_labels)
```

Output:

```
Solving test1
224 letters were correctly labelled which is about 99% of the letters.
Searching for barry
Line from index 192 to index 132
Searching for beardshaw
Line from index 180 to index 68
Searching for bridgeman
Line from index 154 to index 42
Searching for brown
Line from index 92 to index 32
Searching for cane
Line from index 30 to index 75
Searching for crowe
Line from index 55 to index 51
Searching for don
Line from index 186 to index 184
Searching for fish
Line from index 63 to index 111
Searching for flowerdew
Line from index 44 to index 156
Searching for hoare
Line from index 131 to index 191
Searching for jekyll
Line from index 165 to index 90
Searching for jellicoe
Line from index 74 to index 172
Searching for kent
Line from index 217 to index 214
Searching for langley
Line from index 8 to index 2
Searching for nesfield
Line from index 219 to index 107
Searching for paine
Line from index 29 to index 85
Searching for paxton
Line from index 89 to index 164
Searching for peto
Line from index 151 to index 106
Searching for repton
Line from index 91 to index 16
Searching for robinson
Line from index 1 to index 113
Searching for roper
Line from index 22 to index 26
Searching for shenstone
Line from index 223 to index 103
Searching for vanbrugh
Line from index 59 to index 157
Searching for wright
Line from index 39 to index 34
Solved
24 out of 24 found correctly.
```



Jack Cheng Ding Han 150159519 ACA15JCH

Trial 2

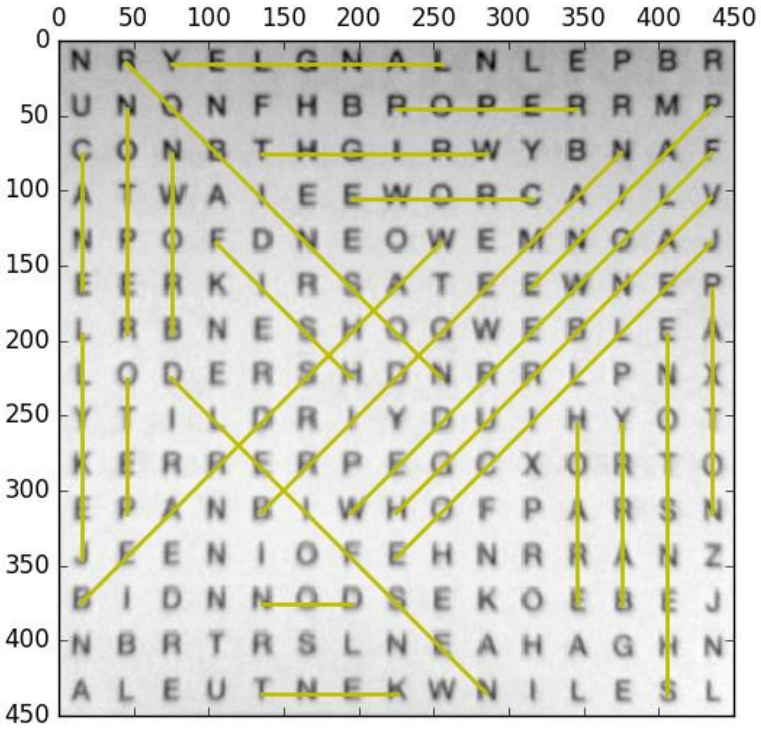
Without dimensionality reduction, the classifier performed better, classifying every word correctly. Every word had an exact match.

Input:

```
wordsearch(test1, words, train_data, train_labels, True, PCA_I)
```

Output:

```
Solving test1 With Reduction
224 letters were correctly labelled which is about 99% of the letters.
Searching for barry
Line from index 192 to index 132
Searching for beardshaw
Line from index 180 to index 68
Searching for bridgeman
Line from index 154 to index 42
Searching for brown
Line from index 92 to index 32
Searching for cane
Line from index 30 to index 75
Searching for crowe
Line from index 55 to index 51
Searching for don
Line from index 186 to index 184
Searching for fish
Line from index 63 to index 111
Searching for flowerdew
Line from index 44 to index 156
Searching for hoare
Line from index 131 to index 191
Searching for jekyll
Line from index 165 to index 90
Searching for jellicoe
Line from index 74 to index 172
Searching for kent
Line from index 217 to index 214
Searching for langley
Line from index 8 to index 2
Searching for nesfield
Line from index 219 to index 107
Searching for paine
Line from index 29 to index 85
Searching for paxton
Line from index 89 to index 164
Searching for peto
Line from index 151 to index 106
Searching for repton
Line from index 91 to index 16
Searching for robinson
Line from index 1 to index 113
Searching for roper
Line from index 22 to index 26
Searching for shenstone
Line from index 223 to index 103
Searching for vanbrugh
Line from index 59 to index 157
Searching for wright
Line from index 39 to index 34
Solved
24 out of 24 found correctly.
```



Jack Cheng Ding Han 150159519 ACA15JCH

Trial 3

With dimensionality reduction, the classifier classifies at least half of the letters correctly and the wordsearch function likewise draws at least half of the lines on the image correctly.

Input:

```
wordsearch(test2, words, train_data, train_labels, True, PCA_I)
```

Output:

```
Solving test2 With Reduction
141 letters were correctly labelled which is about 62% of the letters.
Searching for barry
Line from index 192 to index 132
Searching for beardshaw
Line from index 180 to index 68
Searching for bridgeman
Line from index 154 to index 42
Searching for brown
Line from index 92 to index 32
Searching for cane
Line from index 59 to index 101
Searching for crowe
Line from index 160 to index 220
Searching for don
Line from index 186 to index 184
Searching for fish
Line from index 63 to index 111
Searching for flowerdew
Line from index 44 to index 156
Searching for hoare
Line from index 131 to index 191
Searching for jekyll
Line from index 141 to index 211
Searching for jellicoe
Line from index 116 to index 214
Searching for kent
Line from index 156 to index 198
Searching for langley
Line from index 8 to index 2
Searching for nesfield
Line from index 215 to index 117
Searching for paine
Line from index 29 to index 85
Searching for paxton
Line from index 89 to index 164
Searching for peto
Line from index 116 to index 158
Searching for repton
Line from index 91 to index 16
Searching for robinson
Line from index 1 to index 113
Searching for roper
Line from index 22 to index 26
Searching for shenstone
Line from index 223 to index 103
Searching for vanbrugh
Line from index 59 to index 157
Searching for wright
Line from index 39 to index 34
Solved
17 out of 24 found correctly.
```