Image Recognition Algorithm Analysis

Zach Robinson

May 19, 2016

1 Approach

I took a simple approach to designing this image recognition application. At onset, I felt that this simple approach paired well with the simple nature of the intended images to be classified. From a broad perspective, in order to classify an image, the application takes the input image and compares it against a dictionary of known and classified images. The application includes a function that can be used to build the image reference dictionary. The user of the application can specify what from what images the dictionary is built and includes in the dictionary the known classification of each image. This dictionary is static. If the user wishes to update the dictionary, in the current version, an entirely new dictionary must be created. The application gains no further 'knowledge' with each classification that it performs. In other words, the application does not include any 'learning' capability. The user of the application is responsible for the 'knowledge' base of the application in the form of the reference dictionary.

Each image is processed before classification or addition to the reference dictionary. The first step of the processing is in converting the image file into its matrix/array representation, where each pixel of the image is converted into an [R, G, B] 3-tuple and pixels are arranged in rows according to their place in the image. After the conversion to a matrix, each pixel is run through a threshold function that converts each pixel into either black or white, [0, 0, 0] or [255, 255, 255]. If a pixel in the original image is anything less than pure white ([255, 255, 255]), that pixel is converted into a purely black pixel. This processing is done for the following reasons: for the images provided, there are no colors outside of the black-white spectrum, and even in the event that there were, color is not a significantly important factor for determining the classification of the particular set of images presented. In other words, the crux of the difference between a 'smile' and a 'hash' is not in its color. This thresholding also renders the images more easily comparable for the algorithm. It does not have to distinguish between or compare varying levels of grayness and instead focuses on the difference between blank (white) space and parts of the actual image design. One important thing to note here is that the algorithm assumes a white background with a non-white design. Anything outside of this scope will be unintelligible.

The final aspect of the approach that I will highlight is that of the actual image classification. As stated above, in order to classify an image, the application compares that image to a reference dictionary. The comparison is done over each image of the dictionary. For each individual comparison, the number of pixel matches is recorded. The pixel matches are recorded per classification type. Each classification type carries its own pixel-match total. In other words, if the dictionary image for a given comparison is of classification 1, then any matches found during that comparison are added to the classification 1 total. When the image being classified has been compared against each dictionary element, the match totals are compared. The application chooses the classification with the highest match total as the classification for the image in question. It is important to highlight some limitations of this strategy. Firstly, each classification type must have the same number of images in the reference dictionary so as to avoid bias in the match totals. Secondly, the comparison does not account at all for the spacial relationship between pixel matches found. Because of this, if the dictionary does not contain a variety of orientations of each classification, the application will struggle to classify an image that has been tilted or shifted from the normal position, even if that unaltered image is already an element of the dictionary.

2 Accuracy

A reference dictionary containing 60 images of each classification type was used for testing. Tests were performed using 12 images of each classification type, not included in the reference dictionary. These numbers were chosen primarily due to the size of provided test data sets. There were 72 'Hat' images included in the provided testing data, so I split this set into 60/12 for dictionary/testing and used the same split for each other classification, with some images unused.

The first testing of the application yielded poor overall results, an overall accuracy of 70%. Upon further examination, this poor overall performance could be attributed to a particularly poor recall rating for the 'Hash' classification of 27%. An additional test performed using only the four non-Hash image types yielded a significantly better accuracy rating of 84%.

Following this initial round of testing I decided to make some modifications in order to improve the overall accuracy rating, specifically attempting to bring up the low recall score for the Hash classification. In order to accomplish this, I implemented a weight system. Instead of each matched pixel during comparison counting as 1 unit consistently for each classification type, I first decided to give more weight to each Hash classification match. In other words, while each match for the other classifications counted as 1 tally, the matches for the Hash classification counted for some amount greated than 1. In this way, for any given image, it was more likely for the Hash classification to have the highest tally. It was clear that after testing this initial change, more weights were necessary to offset the Hash weight. I ultimately decided, after various tweaking and testing, to apply weights to the Smile and Heart classifications in addition to the Hash classification. The weights are as follows: Smile = 96/95, Hash = 41/40, Heart = 76/75. After applying this weight system, the following accuracy results were measured:

Overall Accuracy: 83.33%

Smile Recall: 75%
Smile Precision: 75%
Hat Recall: 100%
Hat Precision: 100%
Hash Recall: 75%
Hash Precision: 69.23%
Heart Recall: 75%
Heart Precision: 75%
Dollar Recall: 91.67%
Dollar Precision: 100%

3 Conclusion

After modification and testing over the given data set, one can be reasonably confident in the overall ability of the application to classify similar images. One can be very confident in the accuracy of the classification determined by the application for determinations of 'Hat' and to a slightly lesser extent, 'Dollar,' with less confidence in determinations of 'Smile,' 'Heart,' or 'Hash.' More simply stated, if the application says 'Hat' it's almost surely a hat. If the application says 'Dollar' it's almost surely a dollar. If the application says anything else, it is probably right, but you might want to look at it yourself to make sure.