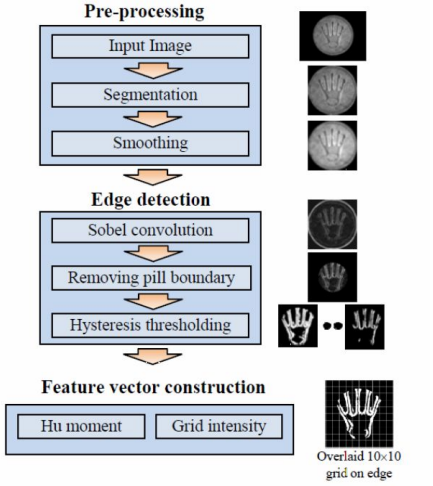
Rajat Bhageria and Kashish Gupta
CIS 581
1 November 2017
Project 3 Part B

Pill Object Classification Using Edge Detection and Chamfer Edge Matching

1. Abstract: Pill detection has the potential to reduce medical and pharmaceutical error, and to make lives easier for all patients. Using edge detection and edge matching, we will create a classification algorithm that matches cropped images of pills to a set of training (sample) images. Using chamfer matching and distance transform, we will compute the "distance" between the test image and each of the training images, and the program will output the classification of the training image which is a closest match (ie. the least distance). Classifying pills can be used in healthcare applications to reduce the stress of managing different pills at different times of the day.

2. Introduction: Pill detection is an immensely important application for people of all ages, especially since children, the elderly, and the visually impaired can take the wrong pill and this can injure or even kill them. In particular, FDA[1] estimates that about 1.3 million people are injured by medication errors annually in the U.S. As such, in this project, we are working to recognize pills using edge matching based on the Canny Edge Detector.

3. Related Works: We were able to find two main papers that have created a similar algorithm to classify pills using edges.



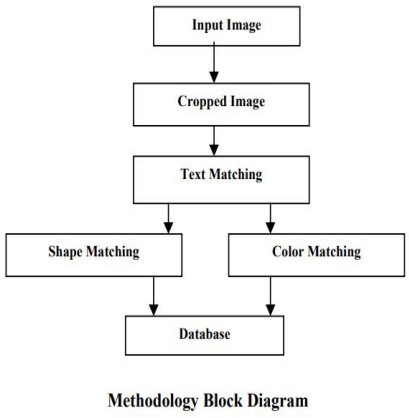


Figure 1 (left): *PILL-ID: Matching and Retrieval of Drug Pill Imprint Images* [2]
Figure 2 (right)*: Color and Shape Feature Extraction and Matching in Pill Identification Systems* [3]

4. Implemented Methodology:

1) Find five different training pills with reasonably different shapes and all in slightly different orientations. It's important that given all 5 images, a human can recognize that they are in fact

[1] https://www.fda.gov/Drugs/DrugSafety/MedicationErrors/ucm080629.htm
[2] Lee, Park, Jain, http://ieeexplore.ieee.org/abstract/document/5595988/
[3] Annasaro, Hema, http://ijcsit.com/docs/Volume%205/vol5issue02/ijcsit2014050211.pdf

different from each other, even if the human doesn't know which kind they each are. Our library had three different versions of all 5 library images (leading to total of 15 library images in the dataset) increasing the probability that our classifier has a high true positive rate.

2) Create a labeled library of all the data. Create an edge map of all the pills using Canny Edge and store all the images in a 5x1 cell array. Similarly, create a 1x5 string matrix and store all the labels corresponding to each respective cell array into it.
3) Create a method that can crop any edge map to the border of the edge map so that the corners of the pill are tangent to the border of the image.
4) Crop each of the edge maps in the library to make sure that the corners of the pill are tangent to the border of the image. This is simply to abstract out the idea that the image could be different sizes or could be in a different position.
5) Scale each of the library images after cropping to a particular dimension. This is to ensure that each of the images are the exact same size.
6) Now take a new query image of a pill and run Canny Edge detector on it. Then crop this image using the cropping class created earlier to ensure that the query pill edge map being compared to labeled pills edge maps have the relatively similar sizes. Then scale the query image to be the same size as each of the images in the library.
7) Find the label in the library that's the closest match to the query image using chamfer matching:
    a) For each of the images in the library:
        i) Return the cost of the difference between the query image and the library images. To do this, overlay the edge map of the query image on top of the edge map of the library image. For each pixel in the query image, find the pixel with value "1" that's closest in position to the library image and euclidian distance between these two pixels. Do this for all pixels in the query image and compute the total cost of all the pixels in the edge map. Divide by number of pixels in the edge map to get the average distance between the pixels in both images. We used dsearchn to do this.
        ii) If the cost of this library image is less than the cost of previous library images, store the cost of this image as the minCost and store the label.
        iii) If the cost of this library image is greater than the cost of previous library images, store the cost of this image as the maxCost. This maxCost will be used to compute the probability measure of how likely our label prediction is to be the correct one.
8) Return the label of the image that corresponds with minCost and return the accuracy as (maxCost - minCost) / maxCost, because this will give us our performance gain over the maxCost.

5. Comparison with your proposed methodology:

There are a few major differences between our proposed methodology and the implemented methodology:

- Distance transform vs Chamfer matching: Before we had thought to use an epsilon value, ε, to compare the query edge map with the library of labeled edge maps. In other words, if pixels in two edge maps are within a distance ε of each other, we'll still consider the pixel to be zero cost. Then we'd find the total cost using a similar methodology.
    - However, this time instead of treating pixels within ε to have a cost of zero, we simply avoided distance transform and found the total cost of each image in the library image using dsearchn. We found the average distance between pixels in the library set to pixels in the query image, looking only at pixels that were closest to each other in position. Finding the average minimum distance between the two images gave us the same effect as a distance transform, using the concepts of chamfer matching.
    - This is a significantly more robust cost function and works well since an epsilon that may work well for a particular image (or a particular class of images) may not work for another class. This classifier abstracts this idea out and returns the optimal label regardless of epsilon.
- A significantly more robust image cropper: In the initial proposal, we were always intending to crop the image but during the implementation because of bugs we faced, our the implemented cropper finds the first instance of 1 closest to the left edge, right edge, top edge, and bottom edge and crops there. If there are any outliers where there's a patch of ones with no lines connecting to it, we disregard that patch and crop over it.
- Adding a scaling factor after cropping: We had not intended to scale images after cropping them in the initial proposal. However, we then noticed that if we don't scale all the library images to the largest dimension of the cropped library image, we couldn't accurately compare to the query image. Thus after cropping, we had to scale our library image.
- Improved the probability measure for our classifier by comparing the cost of our label prediction to the maxCost of the least comparable library image.

6. Achieved results:

Our achieved result is a classifier in which you can feed in images of pills (of any size) but in a particular orientation and our classifier will label the pill for you. In terms of orientation, it's important that the query picture is taken so that 1) circular pills are taken with the words parallel with the top and bottom edges and right-side up and 2) ellipsoidal pills are taken in so that their long edges are parallel with the top and bottom edges.

However because of the limited scope of this project, our classifier only works with a few different kinds of pills (the five we have in our classifier library).

Here are some of the tests we ran that our classifier worked well on. Note that the image are exactly the same as we passed into our classifier (i.e., different sizes and colors and brightness).

Example 1: Correctly identified query image as "Ibuprofen" with 87.6235% probability:

Figure 3 (left): *The query image*
Figure 4 (right): *Our library image associated with the label "ibuprofen"*

Example 2: Correctly identified query image as "Advil C&S" with 91.3037% probability:

Figure 5 (left): *The query image*
Figure 6 (right): *The significantly larger and at a different angle library image associated with the label "Advil C&S"*

Example 3: Correctly identified query image as "Vicodin" with 98.7114% probability:

Figure 7 (left): *The query image*
Figure 8 (right): *Our library image associated with the label "Vicodin"*

Example 4: (Query Image Not in Library) Identified query image as "Accupril" with 59.4949% probability:

Figure 9 (left): *The query image (note that this pill is not in our database!)*
Figure 10 (right): *Our library image associated with the label "Accupril"*

As you can see, our classifier works very well in most cases. We spent significant time tuning the thresholding for edge linking in the Canny Edge Detector but ultimately our classifier was getting 85% probability rates for true positives.

As Figure 9 and 10 show, our classifier will always return a result but of course the probability is quite small at <60%. In other words, we leave it up to the user to determine based on the probability to trust or not to trust the returned label. A future feature might be that pill recognition is such a potentially hazardous task, if the accuracy is less than 85%, we return a warning message to the user expressing this: "We're not very confident in our results and you should ask an expert before consuming or testing this pill."

7. Analysis of comparison with expected results:

We expected that the shape of the pill would be the primary determining factor in classifying the type of pill, but found actually that edge detection was especially good at finding edges along the labels of the pills. These labels contributed most to decreasing the cost of the most similar images.

We also expected that shadows would prevent correct classification, but even with shadows (as displayed above), the images were correctly classified.

Lastly, we did not think the background of the pill would contribute to so much noise, but during edge detection, the background of the pill added unwanted pixels to the edge map, therefore we had

to change the threshold of edge linking in order to remove background edges. This was especially relevant while cropping the image, because we need to identify the foreground vs. background to crop the image correctly. Having a standard cropping for all images is the reason we did not need to translate the query image over the training image during chamfer matching.

8. Future work and references:

This project has immense applications in the fields medical care and criminology. It is well known that patients struggle with understanding the differences between the many pills they are prescribed during terminal illness or post-surgery, and this sort of algorithm would allow a simple smartphone app to classify pills for them. In pharmaceuticals, this software would reduce error and have the ability to sort pills without the need for a human. It is possible that with future work it could be be improved to detect a partial pill or just fragments of a pill, which would be useful in forensics, especially regarding illegal drugs.

In the future, we would like to add a few more different kinds of matching (in addition to edge matching) to recognize pills correctly. In particular we would like to add:

- Color matching
- Optical character recognition to read text
- Texture matching to ensure that the query pill has a similar texture to the sample pill
- Lumosity matching to make sure that light is being reflected in similar ways
- Have a much larger library of labeled pills to ensure we can make the most precise classification.
- Orientation standardizing. One of the main problems with the current classifier is that it requires both the query and the library images to be in a somewhat similar orientation. In the future it would be ideal to create a classifier that can also standardize for orientation when doing the chamfer edge matching.

9. Contributions of each member:

- Rajat: Overall architecture of the code and MATLAB scripts, cropping and scaling images, canny edge detection and thresholds, writing final report, writing proposal
- Kashish: Finding the training and query images, script for importing images into MATLAB, cost function for finding minimum distance pixels between images, writing final report, writing proposal