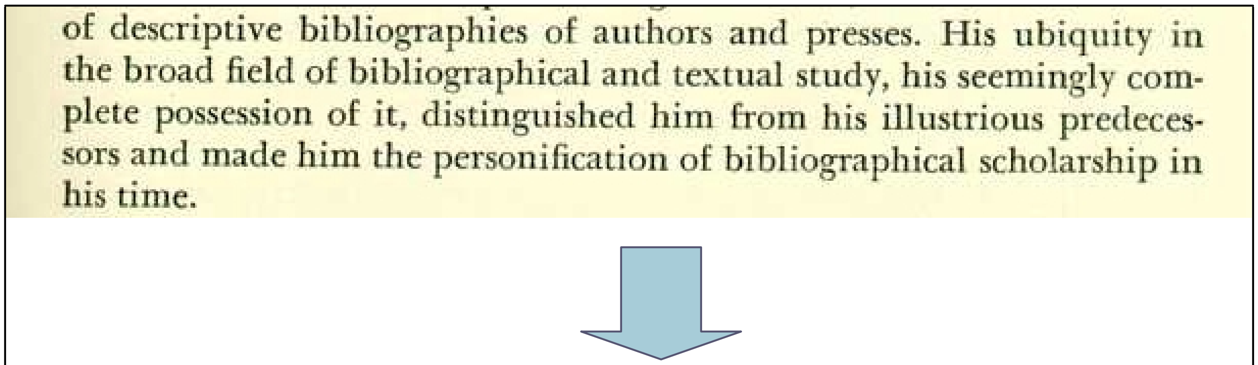
A Matlab Project in Optical Character Recognition (OCR)

Jesse Hansen

Introduction: What is OCR?

The goal of Optical Character Recognition (OCR) is to classify optical patterns (often contained in a digital image) corresponding to alphanumeric or other characters. The process of OCR involves several steps including segmentation, feature extraction, and classification. Each of these steps is a field unto itself, and is described briefly here in the context of a Matlab implementation of OCR.

One example of OCR is shown below. A portion of a scanned image of text, borrowed from the web, is shown along with the corresponding (human recognized) characters from that text.



of descriptive bibliographies of authors and presses. His ubiquity in the broad field of bibliographical and textual study, his seemingly complete possession of it, distinguished him from his illustrious predeces-sors and made him the personification of bibliographical scholarship in his time.

of descriptive bibliographies of authors and presses. His ubiquity in the broad field of bibliographical and textual study, his seemingly com-plete possession of it, distinguished him from his illustrious predeces-sors and made him the personification of bibliographical scholarship in his time.

Figure 1: Scanned image of text and its corresponding recognized representation

A few examples of OCR applications are listed here. The most common for use OCR is the first item; people often wish to convert text documents to some sort of digital representation.

1. People wish to scan in a document and have the text of that document available in a word processor.
2. Recognizing license plate numbers
3. Post Office needs to recognize zip-codes

Other Examples of Pattern Recognition:

1. Facial feature recognition (airport security) – Is this person a bad-guy?
2. Speech recognition – Translate acoustic waveforms into text.
3. A Submarine wishes to classify underwater sounds – A whale? A Russian sub? A friendly ship?

The Classification Process:

(Classification in general for any type of classifier) There are two steps in building a classifier: training and testing. These steps can be broken down further into sub-steps.

1. **Training**

   a. *Pre-processing* – Processes the data so it is in a suitable form for…
   b. *Feature extraction* – Reduce the amount of data by extracting *relevant* information—Usually results in a vector of scalar values. (We also need to NORMALIZE the features for distance measurements!)
   c. *Model Estimation* – from the finite set of feature vectors, need to estimate a model (usually statistical) for each class of the training data

2. **Testing**

   a. *Pre-processing*
   b. *Feature extraction* – (both same as above)
   c. *Classification* – Compare feature vectors to the various models and find the closest match. One can use a distance measure.

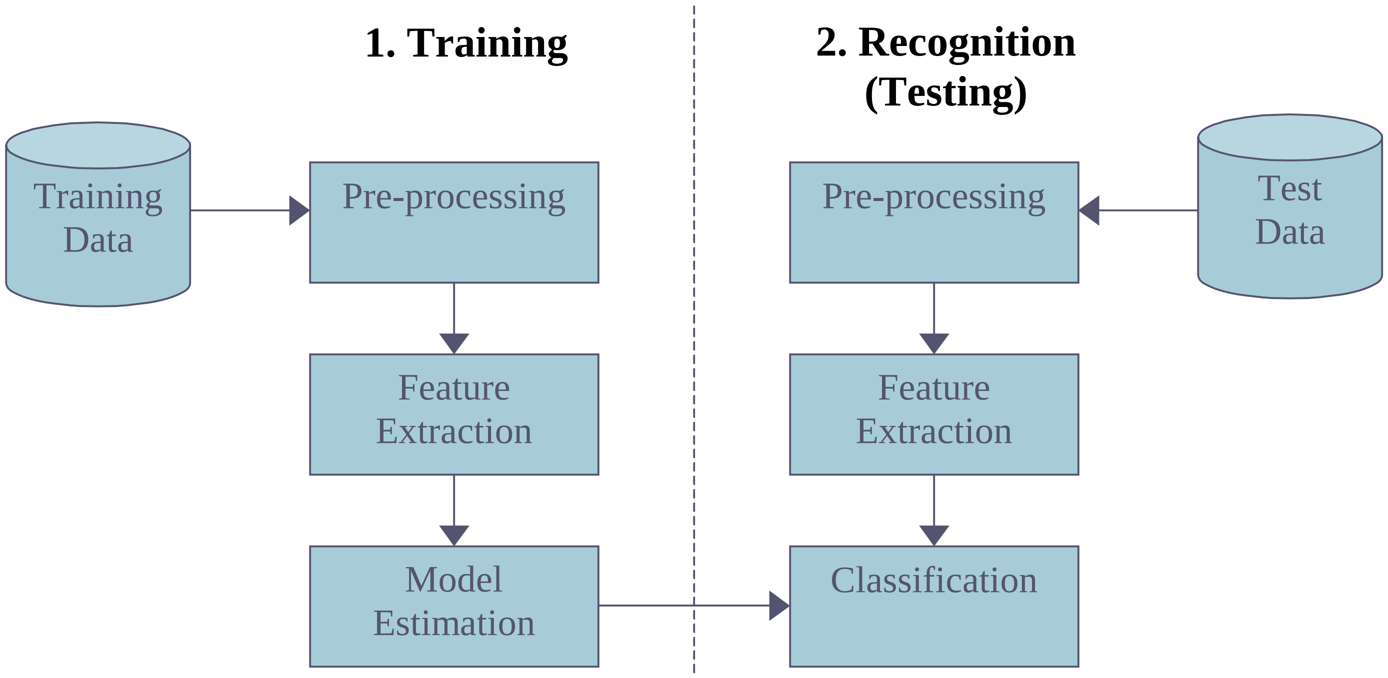


Figure 2: The pattern classification process

OCR – Pre-processing

These are the pre-processing steps often performed in OCR

Binarization – Usually presented with a grayscale image, binarization is then simply a matter of choosing a threshold value.

Morphological Operators – Remove isolated specks and holes in characters, can use the *majority* operator.

Segmentation – Check connectivity of shapes, label, and isolate. Can use Matlab 6.1's *bwlabel* and *regionprops* functions. Difficulties with characters that aren't connected, e.g. the letter *i*, a semicolon, or a colon (; or :).

Segmentation is by far the most important aspect of the pre-processing stage. It allows the recognizer to extract features from each individual character. In the more complicated case of handwritten text, the segmentation problem becomes much more difficult as letters tend to be connected to each other.

OCR – Feature extraction (see reference [2])

Given a segmented (isolated) character, what are useful features for recognition?

1. Moment based features

   Think of each character as a pdf. The 2-D moments of the character are:

   $$m_{pq} = \sum_{x=0}^{W-1}\sum_{y=0}^{H-1} x^p y^q f(x,y)$$

   From the moments we can compute features like:

   1. Total mass (number of pixels in a binarized character)
   2. Centroid - Center of mass
   3. Elliptical parameters
      i. Eccentricity (ratio of major to minor axis)
      ii. Orientation (angle of major axis)
   4. Skewness
   5. Kurtosis
   6. Higher order moments

2. Hough and Chain code transform
3. Fourier transform and series

OCR - Model Estimation (see reference [1])

Given *labeled* sets of features for many characters, where the labels correspond to the particular classes that the characters belong to, we wish to estimate a statistical model for each character class. For example, suppose we compute two features for each realization of the characters 0 through 9. Plotting each character class as a function of the two features we have:

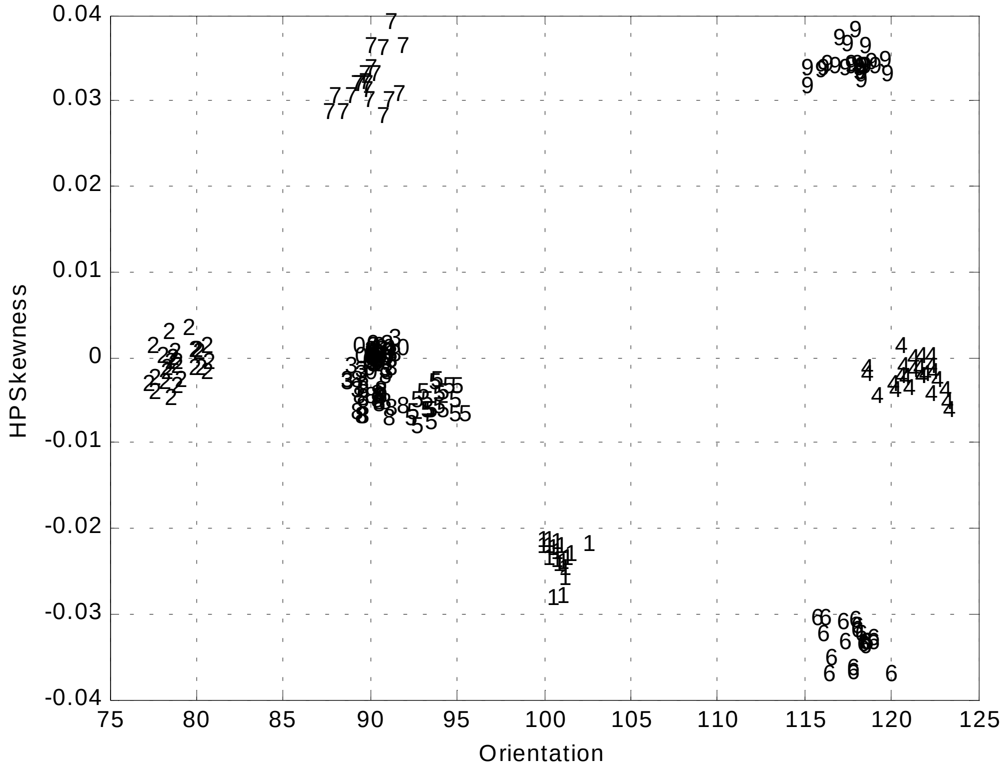


Figure 3: Character classes plotted as a function of two features

Each character class tends to cluster together. This makes sense; a given number should look about the same for each realization (provided we use the size font type and size). We might try to estimate a pdf (or pdf parameters such as mean and variance) for each character class. For example, in Figure 3, we can see that the 7's have a mean Orientation of 90 and HPSkewness of 0.033.

OCR – Classification (see reference [1])

According to Tou and Gonzalez, "The principal function of a pattern recognition system is to yield decisions concerning the class membership of the patterns with which it is confronted." In the context of an OCR system, the recognizer is confronted with a sequence feature patterns from which it must determine the character classes.

A rigorous treatment of pattern classification is beyond the scope of this paper. We'll simply note that if we model the character classes by their estimated means, we can use a distance measure for classification. The class to which a test character is assigned is that with the minimum distance.

The Matlab Implementation:

The Character Classifier Graphical User Interface (GUI)

A Matlab GUI was written to encapsulate the steps involved with training an OCR system. This GUI permits the user to load images, binarize and segment them, compute and plot features, and save these features for future analysis. The file is called *train.m*, and is available at:

http://www.uri.edu/~hansenj/projects/ele585/OCR/

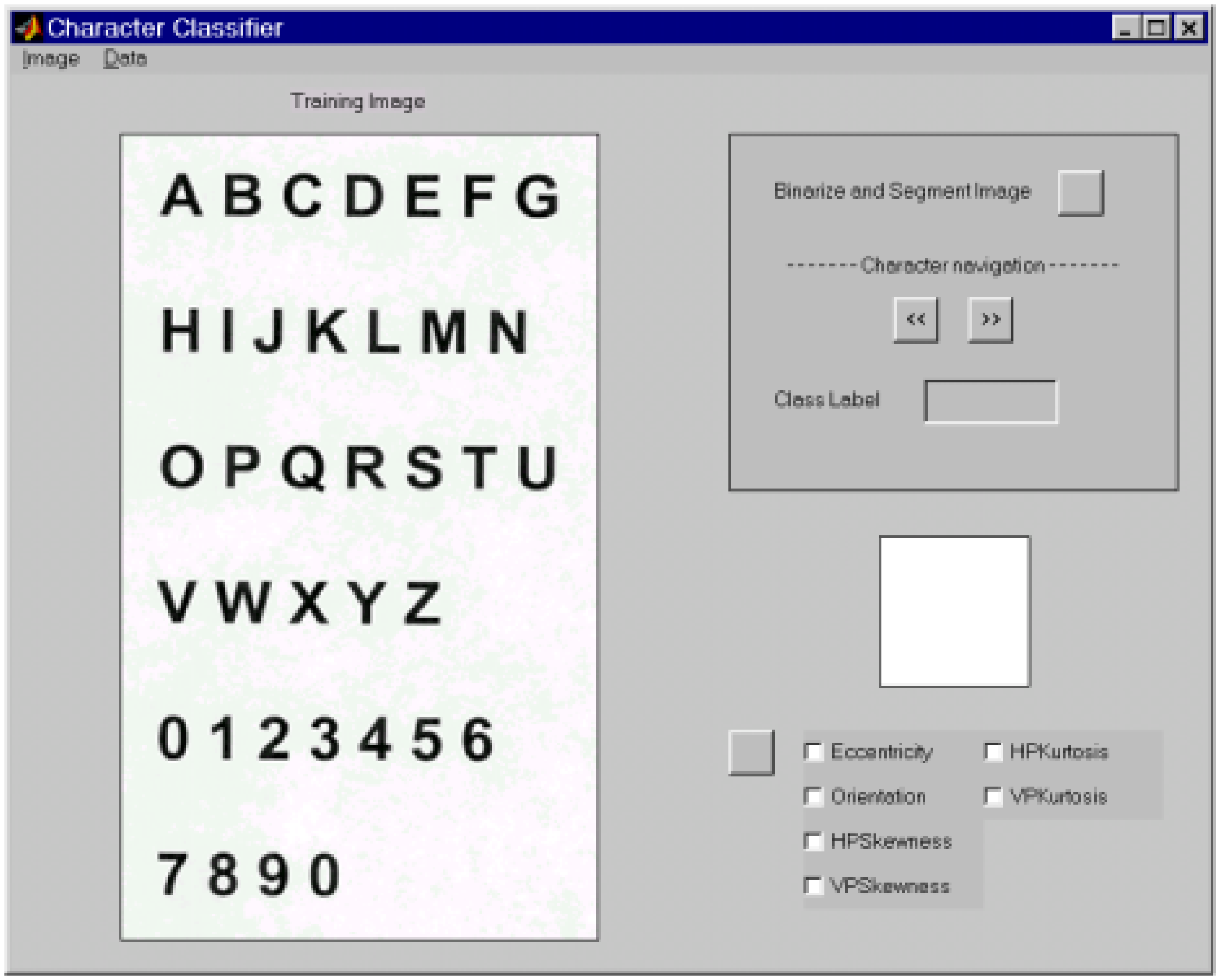


Figure 4: The Character Classifier Graphical User Interface

Loading an Image

Images can be imported into the GUI by clicking on the **Image** menu and selecting **Open**. Both TIF and JPG file formats are supported. Most of the testing was done with grayscale TIF images (with no LZW compression).

Binarize and Segment

After opening an image, it can be converted to black and white and segmented by clicking in the button in the upper right corner of the window (see Figure 4). This button will also extract the various features.

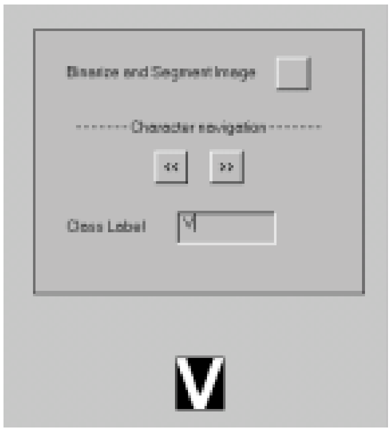

Binarize and Segment Image
-------- Character navigation --------
<<
>>
Class Label
V

Figure 5: Labeling the characters.

Labeling the Characters

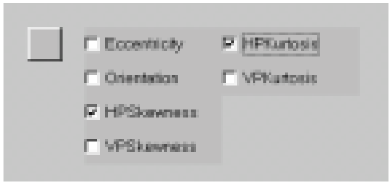
Once the training image is segmented, a character will appear below the text box titled **Class Label**. It's the user's job to label each segmented character appropriately. Once a character label has been entered into the text box, click "**>>**" to move to the next character. One can navigate back by clicking "**<<**".

Saving and Loading the Features, Labels, etc.

Segmented images, character features, and labels can be saved by clicking on the **Data** menu and selecting **Save**. The characters need not be labeled for data saving to occur. Load image data (features, etc.) by clicking on the **Data** menu and selecting **Load**. See Figure 4.

Plotting Class/Features Information

All the characters must be labeled before class/feature information can be plotted. If the characters are labeled, select two of the features by checking the appropriate boxes. Next, click on the unlabeled button to plot the characters classes as a function of the features. If more than two boxes are checked, only the first two selected features will be used.



☐ Eccentricity
☐ Orientation
☑ HPSkewness
☐ VPSkewness
☑ HPKurtosis
☐ VPKurtosis

Figure 6: Select features to plot

References

[1] J.T. Tou and R.C. Gonzalez, *Pattern Recognition Principles*, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1974

[2] M. Szmurlo, Masters Thesis, Oslo, May 1995, (users.info.unicaen.fr/~szmurlo/papers/masters/master.thesis.ps.gz)