

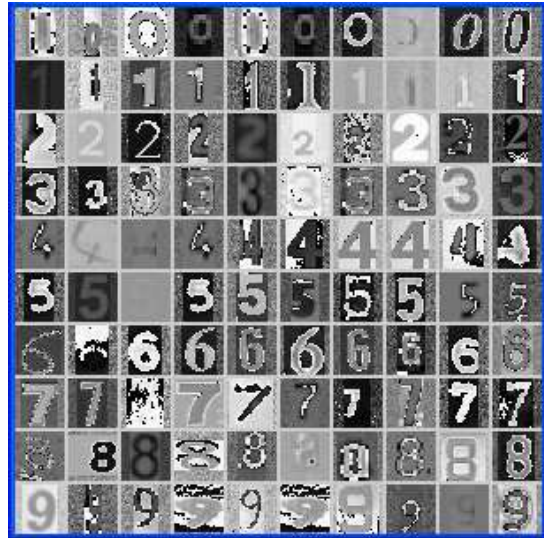
CSC384 Assignment 3 : Character Recognition

Due Date: Friday, December 8th @ 11:59:59 AM

Introduction

Vision and perception are difficult tasks, even for humans. In fact, it is a requirement that doctors have handwriting that is completely illegible, and are required to take a special course in obfuscating their penmanship if anybody other than a pharmacist can tell what they've written¹.

An illustration of this problem is the task of character recognition. No matter what font one uses in a document, any human who reads the document should be able to identify and distinguish between the various characters. The human brain does this by combining its knowledge of the distinctive features of past characters with the observed features of the current character, to determine what the most likely category would be for the character being observed.



In this assignment, you will be implementing a basic character recognizer in Matlab to generate and identify various numerical digits, using simple probability models.

The Character Datasets

Before you begin, go to the course web page, and download the following Matlab files:

- `model.mat`
- `show.m`
- `data.mat`

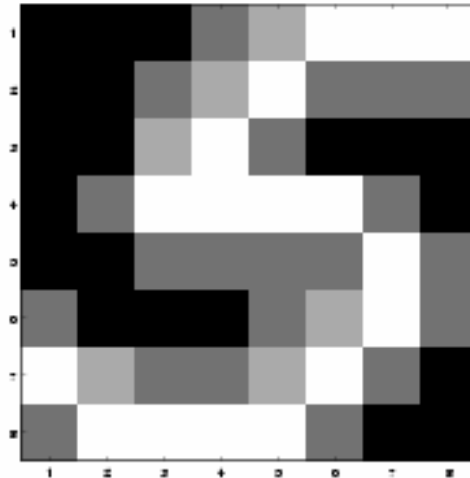
In Matlab, type `load model.mat`. This will load the arrays `digits` ($8 \times 8 \times 10$) and `noise` (1×7). The array `digits` contains image templates for the digits $\{1, \dots, 9, 0\}$. To see them, type `show(digits(:, :, d))` for $d = 1, \dots, 10$. There is also a collection of real handwritten digits in the file `data.mat`, which contains 1100 examples of each digit, in an $8 \times 8 \times 1100 \times 10$ array called `data`.

¹ NOTE: This fact may have been completely fabricated by the instructor of this course.

Background

In this assignment, digits images will be represented by 8 x 8 arrays of greyscale numbers, where 1 = black, 2 = dark grey, 3 = light grey and 4 = white. For example, the following matrix represents the corresponding image:

$$\begin{bmatrix} 1 & 1 & 1 & 2 & 3 & 4 & 4 & 4 \\ 1 & 1 & 2 & 3 & 4 & 2 & 2 & 2 \\ 1 & 1 & 3 & 4 & 2 & 1 & 1 & 1 \\ 1 & 2 & 4 & 4 & 4 & 4 & 2 & 1 \\ 1 & 1 & 2 & 2 & 2 & 2 & 4 & 2 \\ 2 & 1 & 1 & 1 & 2 & 3 & 4 & 2 \\ 4 & 3 & 2 & 2 & 3 & 4 & 2 & 1 \\ 2 & 4 & 4 & 4 & 4 & 2 & 1 & 1 \end{bmatrix}$$

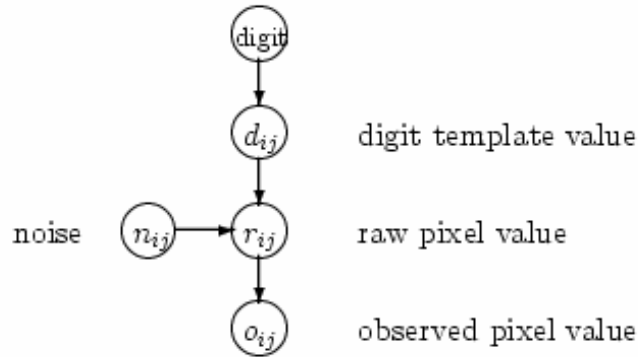


The goal is to implement a probability model that can be used to generate and recognize images of handwritten digits. In addition, for bonus marks, you are given the opportunity to supplement your basic solution by creating your own recognizer.

Part 1: Character Generation

First, we have to generate our observed characters. This is done by adding noise to our character sets, to simulate the variations in viewing a particular character. To do this, we perform the following actions:

1. Pick a digit randomly from among $\{1, \dots, 9, 0\}$ and retrieve its corresponding image template D (an 8 x 8 array).
2. Given the digit template, each image pixel o_{ij} is generated independently by:
 - a. picking a random noise value $n_{ij} \in \{-3, -2, -1, 0, 1, 2, 3\}$ (according to the distribution found in noise),
 - b. adding the noise to the corresponding digit template pixel d_{ij} to obtain the raw pixel value $r_{ij} = d_{ij} + n_{ij}$,
 - c. thresholding the raw value to obtain the observed image pixel.



The threshold value for the observed pixel is calculated in the following manner:

$$o_{ij} = \begin{cases} r_{ij} & \text{if } 1 \leq r_{ij} \leq 4 \\ 1 & \text{if } r_{ij} < 1 \\ 4 & \text{if } r_{ij} > 4 \end{cases}$$

Once this is done, write a Matlab function “gen” that takes two arguments, digits and noise, and returns a random image of a digit generated by the model outlined above.

Part 2: Character Recognition

Given an observed image O , we wish to compute the most likely digit given the image. This amounts to computing the most likely digit image template D given O (since digits and their image templates are in a one-to-one correspondence). Thus, we wish to find:

$$\begin{aligned}
 D^* &= \arg \max_D P(D|O) \\
 &= \arg \max_D P(DO) \\
 &= \arg \max_D \sum_N P(DNO) \\
 &= \arg \max_D \sum_N P(D)P(N)P(O|DN) \\
 &= \arg \max_D P(D) \sum_{n_{11}=-3}^3 \cdots \sum_{n_{88}=-3}^3 \left[\prod_{i=1}^8 \prod_{j=1}^8 P(n_{ij}) \right] \left[\prod_{i=1}^8 \prod_{j=1}^8 P(o_{ij}|d_{ij}, n_{ij}) \right] \\
 &= \arg \max_D P(D) \prod_{i=1}^8 \prod_{j=1}^8 \sum_{n_{ij}=-3}^3 P(n_{ij})P(o_{ij}|d_{ij}, n_{ij})
 \end{aligned}$$

where:

$$P(o_{ij}|d_{ij}, n_{ij}) = \begin{cases} 1 & \text{if } d_{ij} + n_{ij} = o_{ij} \\ & \text{or } o_{ij} = 1 \text{ and } d_{ij} + n_{ij} < 1 \\ & \text{or } o_{ij} = 4 \text{ and } d_{ij} + n_{ij} > 4 \\ 0 & \text{otherwise} \end{cases}$$

For each digit template, $D^{(1)}, \dots, D^{(9)}, D^{(0)}$, we can then compute the conditional probability of $D^{(d)}$ given O by $P(D^{(d)}|O) = P(D^{(d)}O) / \sum_{d=1, \dots, 9, 0} P(D^{(d)}O)$.

Once this is done, write a Matlab function called “`rec`” that takes in three arguments, `obs`, `digits`, and `noise`, where `obs` is an 8 x 8 image of an observed digit, and returns `class` and `probs`, where `class` is the digit $\{1, \dots, 9, 0\}$ that is most likely given `obs`, and `probs` is a 1 x 10 array of the conditional probabilities of each digit given `obs`. The classification and conditional probabilities are to be calculated using the procedures outlined above.

Bonus: Your Own Model

Design a probability model that extends the model mentioned above, and achieves better recognition performance on the test examples in data. Note that the 1100 images in data form a very large set and it would probably take too long to test your model on every image. Instead, you can (repeatedly) test the model on 25 randomly-chosen images for each digit. (The TA will apply a similar test to verify that your model is indeed an improvement).

Include details of your improved model in your report, and submit your recognizer in a function called “`rec2`” for consideration.

Deliverables

Submit the following Matlab files, so that they produce the behaviour specified above:

- `gen.m` – the file that implements the character-generating function
- `rec.m` – the file that implements the character-recognition function
- [`rec2.m` – the (optional) file that implements the character-recognition function]

These files are submitted electronically at the following site:

<http://www.utm.utoronto.ca/submit>

In addition to the electronic submission, a short report (max 3 pages, 12pt font, double-spaced) will also be handed in **at noon on Friday, at my office (SE 4063)**. Grace day submissions may be slipped under the door. The report will describe any design considerations that you want to mention to the marker for this assignment, as well as ideas for improving the recognition module, whether you perform the implementation for it or not.

Mark Breakdown

- 70% Correctness
 - correct operation, not necessarily accurate recognition
- 10% Style & Documentation
- 20% Report (Technique & Analysis)
- 20% Bonus (for those who implement `rec2` well)

Hints and Tips

- Matlab is an easy language, similar to Octave, which has been used in the past. There are plenty of online resources that will help you sail through the syntax hurdles of the language
- Expect that most of the conditional probabilities from the recognizer will be at or near zero or one