Computer Vision – Programming Project 1

王順興　0210184

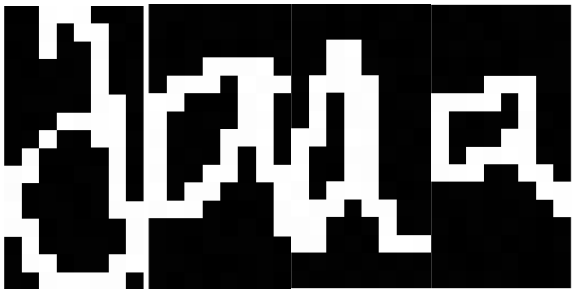
Introduction

Originally, I was going to work on Gesture Recognition. However, the dataset consists of various size of photo taken in various background. At that time, the solution to the issue regarding overflow of det(sigma) is not yet known to me, and the scale of the interest (hand) is another complex problem, so I decided to work on a easier problem, Letter Recognition.

I will be using Matlab to implement this project.

i. The Dataset

The dataset I am using can be found here(http://ai.stanford.edu/~btaskar/ocr/). Each sample is a 16x8 binary image (0 and 1) of an English alphabet (letter) taken from a written word (as opposed to test subjects writing individual letters), this makes the recognition more difficult.



Some sample of 'a'

ii. The Goal

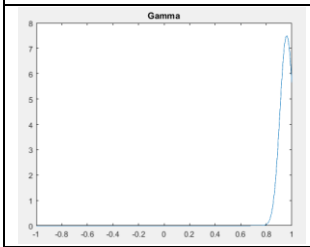
In this project, I hope to be able to classify a given letter to one of 26 classes. Originally, I was going to implement Gaussian MLE and MoG, then compare the two results. But, when 賈恩宇 told me that he found a paper regarding the usage of Bernoulli distribution in binary image, I wondered if Gamma distribution will be a good distribution to fit an image.

And so I'll be comparing the letter classification results of Gaussian MLE and Gamma MLE.

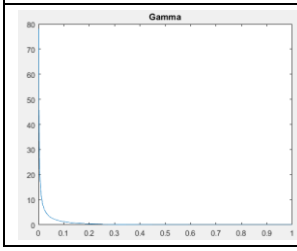
iii. Why Gamma Distribution?

Because Gamma is more flexible; Gamma can do whatever Gaussian can, and some times better!

Fitting of Gaussian noise w/ mu=1, sigma=0.1



Fitting of Gaussian noise w/ mu=0, sigma=0.1



This makes me more curious as to why Gamma Distribution aren't used as much as Gaussian.

Method

i. Gaussian Maximum Likelihood

Gaussian Maximum Likelihood is pretty straight forward, since its means and covariances have closed form:

$$\hat{\mu} = \frac{1}{N}\sum_{n=1}^{N} x_n \quad \text{and} \quad \hat{\sigma^2} = \frac{1}{N}\sum_{n=1}^{N}(x_n - \hat{\mu})^2$$

Compare the posteriors generated by each class to classify a sample:

$$Pr(w=1|\mathbf{x}) = \frac{Pr(\mathbf{x}|w=1)Pr(w=1)}{\sum_{k=0}^{1} Pr(\mathbf{x}|w=k)Pr(w=k)}$$

ii. Gamma Maximum Likelihood

Gamma Distribution, similar to Gaussian, has two parameters: k, θ

$$\frac{1}{\Gamma(k)\theta^k} x^{k-1} e^{-\frac{x}{\theta}}$$

The derivation of these parameters ML fitting can be found in *Estimating a Gamma distribution* by Thomas P. Minka, but his derivation is done in a and b, as in the alternate form of Gamma, so these results are taken from Wikipedia.

θ does have a closed form, though it releies on k:

$$\hat{\theta} = \frac{1}{kN}\sum_{i=1}^{N} x_i$$

k does not have a closed form. It can be found by Newton's method:

Let

$$s = \ln\left(\frac{1}{N}\sum_{i=1}^{N} x_i\right) - \frac{1}{N}\sum_{i=1}^{N}\ln(x_i)$$

the initial approximation of k can be found:

$$k \approx \frac{3 - s + \sqrt{(s-3)^2 + 24s}}{12s}$$

We can update it by Newton's method:

$$k \leftarrow k - \frac{\ln(k) - \psi(k) - s}{\frac{1}{k} - \psi'(k)}$$

From the paper aforementioned, k converges in about 4 iterations.

What about likelihood function? As the ML parameters are dervied from univaraite Gamma Distribution, the likelihood function of a sample is simply the multiplication of univaraite Gamma of each pixel.

$$Likelihood:\ p(\boldsymbol{x}|\boldsymbol{w}) = \prod_{d}^{D} p(x_d|\boldsymbol{w})$$

I did search for a multivariate form of Gamma Distribution, none can be found, and the math required to derive one are simply beyond me. But apart from the issue of having to multiply multiple terms, the accuracy is not being affected by the lack of a multivariate form; as in the Multivaraite Gaussian, we still only use diagonal terms of covariance matrix.

iii. Technical Details

When calculating the Gaussian Likelihood function:

The underflow/overflow issue when calculating Likelihood function is mitigated by using smaller data dimension and normalizing the range from 0 ~ 255 to 0 ~ 1.

The determinant of covariance matrix(sigma) will become 0 when one pixel has the same value across all dataset. This can be solved by adding some noise to each image.

When calculating $\mathbf{k}$ in Gamma Maximum Likelihood:

Notice the form of s

$$s = \ln\left(\frac{1}{N}\sum_{i=1}^{N} x_i\right) - \frac{1}{N}\sum_{i=1}^{N}\ln(x_i)$$

The nature log of each pixel is calculate. What is ln(0)? God knows. So add a tiny value (I chose 0.001) to each pixel that equals to 0.

The size of each class in the dataset is different:

This is very trivial, but anyway this is the size of each class in my dataset:

```
[4034 1284 2114 1442 4955 921 2472 861 4913 189 909 3140 1602 5024 3897 1377 341 2673 1394 2136 2562 664 520 413 1221 1094]
```

Of all 52k samples: 4k are ‘a’, 1.2k are ‘b’… The largest set has 5k sample while some only have few hundreds. So when storing these data, use cell instead of matrix.

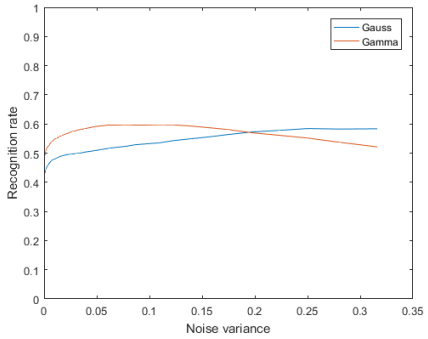
Experiment

The goal is to compare the accuracy of Gaussian and Gamma MLE, and try to find out what causes the difference.

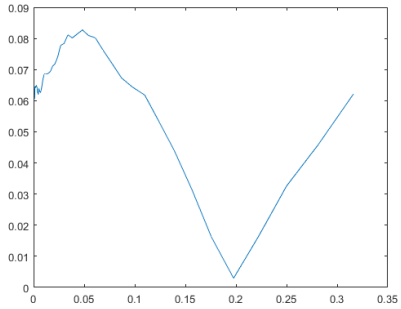
In this comparison, half of the data from each class are used in training; the min(remaining, 200) are used in testing.

For each test, Gaussian Noise is applied on every image. The noise has a mean of 0 and variance from 0.001 to 0.3. Each pixel is limited between 0 and 1 after the noise is applied.

Both estimator assumes that each pixel are independent.



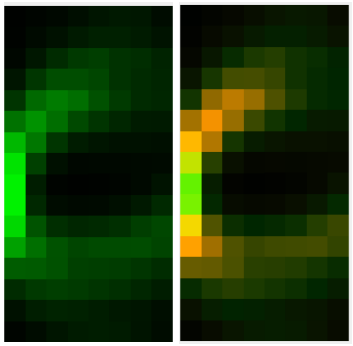
Recognition rate vs Noise Variance



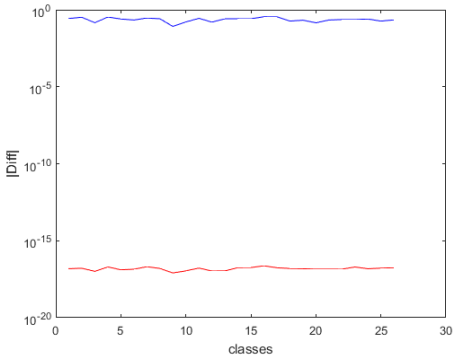
Absolute Diff. in recog. rate vs Noise variance

The first thing I compare is the Recognition rate. When there is little noise, Gamma distribution has a higher recognition rate. But as the noise increases, Gaussian becomes a better estimator.

This makes me wonder what is causing this difference.



Mean Diff. Var. Diff.



←

Red: Diff. Mean

Blue: Diff Var.

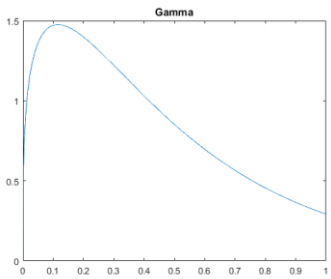
The Diff. Mean is low whatsoever.

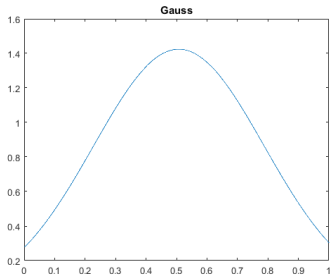
The Diff. Var. is the main factor of difference.

↑ The green part is the average mean of the two estimation. The oranage part shows the difference.

There is no difference in mean, and the difference in variance occurs on pixels with higher variance. So the difference between Gamma and Gaussian might be the way they model variance.

In the **Why Gamma Distribution?** part of this report, I stated that Gamma can do whatever Gaussian can do, and showed two examples. But now we see that Gamma did a poor job when the noise increases, so I did more test.





Fitting of Gaussian noise w/ mu=0.5, sigma=0.3

There are more tests that can't fit into 4 pages. But just by looking at one example with a high variance(noise), we can tell that Gamma has a different fitting strategy. And since the noise that I applied to the data is a Gaussian noise, Gamma has a trouble fitting them when the variance is large.

Maybe this is why Gamma is not used in real world. There is only a few usage of Gamma distribution in Image Processing.

Gaussian noise appears everywhere, and a distribution that cannot properly fit it is therefore useless.