Novel Classifier for Smart Traffic Network

Submitted June 2020, in partial fulfillment of
the conditions for the award of the degree **Bachelor of Computer Engineering/Computer Science.**

Thomas Courtney

3175353

Supervised by Kaushik Mahata ddd

School of Engineering

University of Newcastle

I hereby declare that this dissertation is all my own work, except as indicated in the text:

Signature _____

Date _____ / _____ / _____

Contents

1 Literature Review **1**

1.1 Digital Images 1

1.2 Linear Filtering 3

1.2.1 Correlation 4

1.2.2 Convolution 6

1.3 The Gaussian 7

1.4 Clustering 7

1.4.1 K-Means 8

1.4.2 Mixtures of Gaussians 10

1.4.3 Background Subtraction 12

1.4.4 Expectation Maximization 12

2 Design **13**

Bibliography **13**

List of Tables

List of Figures

1.1 (a) A Simple 8-bit Image (b) Array Representation of Image. 2

1.2 Surface Plot of Figure 1.1a 2

1.3 General Form of a Linear Filter 3

1.4 Visualization of a Filter Kernel Application 3

1.5 Box Filter application and kernel. 4

1.6 Sobel Filters 5

1.7 Application of Sobel filters to exagerate lines. 5

1.8 Correlation of a filter and an image. 6

1.9 Convolution of a filter and an image. 7

1.10 Segmentation of toy cars using K-Means Clustering. 8

1.11 K-Means clustering performed on random data. 10

1.12 Formulation of Mixture of Gaussians. 11

1.13 Individual Gaussians with scaled weightings. 11

Chapter 1

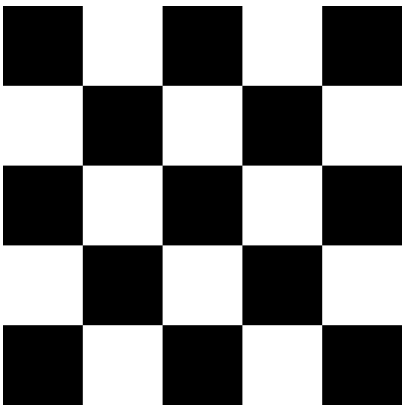
Literature Review

This section of the report expands upon the theory behind the design of the system. Digital image processing and computer vision are broad fields and so only concepts relevant to the system are explored.

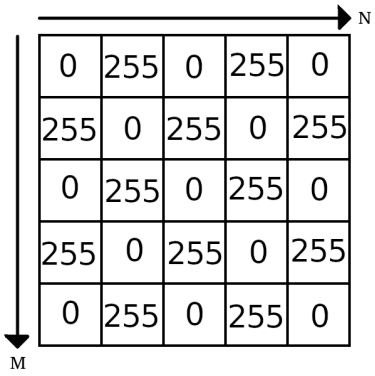
1.1 Digital Images

Digital images consists of a discrete number of discrete valued data points called pixels. It is the value of these pixels that determine their color. A pixel's value is also called its *intensity* and will lie within a range of values determined by the image's encoding. For example an 8-bit image has pixel values in the range $[0, 255]$.

Digital images are often represented as a 2D array of values where each cell corresponds to an intesnity. In software in particular, digital images are stored and manipulated as arrays. An image's array is $M \times N$ pixels in size, which is also its resolution. Notice that in Figure 1.1 the lowest value is 0 (black) and the highest value is 255 (white). All other colors that can be represented by 8-bits are between these two values.



(a)



(b)

Figure 1.1: (a) A Simple 8-bit Image (b) Array Representation of Image.

2D array representation allows pixels to be referenced by their relative position in the M and N directions. Furthermore, M and N may be substituted for the x and y axes of the Cartesian plane. In fact, a digital image is just a two dimensional function where each pixel is described by a coordinate (x,y) and an intensity (z) at that point.

$$z = f(x, y) \tag{1.1}$$

This means that two dimnesional operations may be applied to a digital image and images may be modelled as 3D surfaces. Modelling images as a surface is useful when trying to visualize rates of change in an image. In Figure 1.1 you can see that there is a rapid change in value between different coloured squares, shown as steep cliff edges.

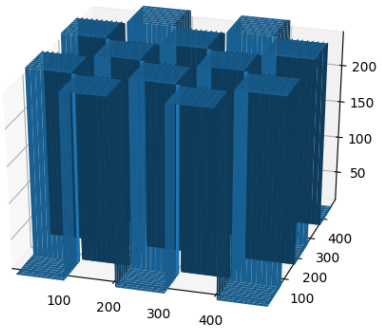


Figure 1.2: Surface Plot of Figure 1.1a

1.2 Linear Filtering

Filtering an image is a way to augment or extract information from it. Linear filters are a subset of operations that operate on a neighbourhood of pixels in an image. All linear filters operate on the same prinicpal, that is, they output a weighted average of their input. Filters take pixels in the neighbourhood covered by the filter as input. It is the weightings of a filter, known as filter coefficients, that determine the effect of the filter. These weights are stored in a matrix called a mask or kernel (see Figure 1.3) that's then convolved or correlated with an image (see sections 1.2.1 & 1.2.2).

$$Kernel = \frac{1}{\sum_{i=0}^{M}\sum_{j=0}^{N} c_{i,j}} \begin{bmatrix} c_{0,0} & c_{0,1} & \dots & c_{0,n} \\ c_{1,0} & c_{1,1} & \dots & c_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m,0} & c_{m,1} & \dots & c_{m,n} \end{bmatrix}$$

Figure 1.3: General Form of a Linear Filter

A filter kernel is nearly always square so as to have a center cell which sits atop a reference pixel. The result of the filter's application at that reference pixel will be stored in the output image at the location of the reference pixel. Notice in Figure 1.4 how the mask sits over the reference pixel.

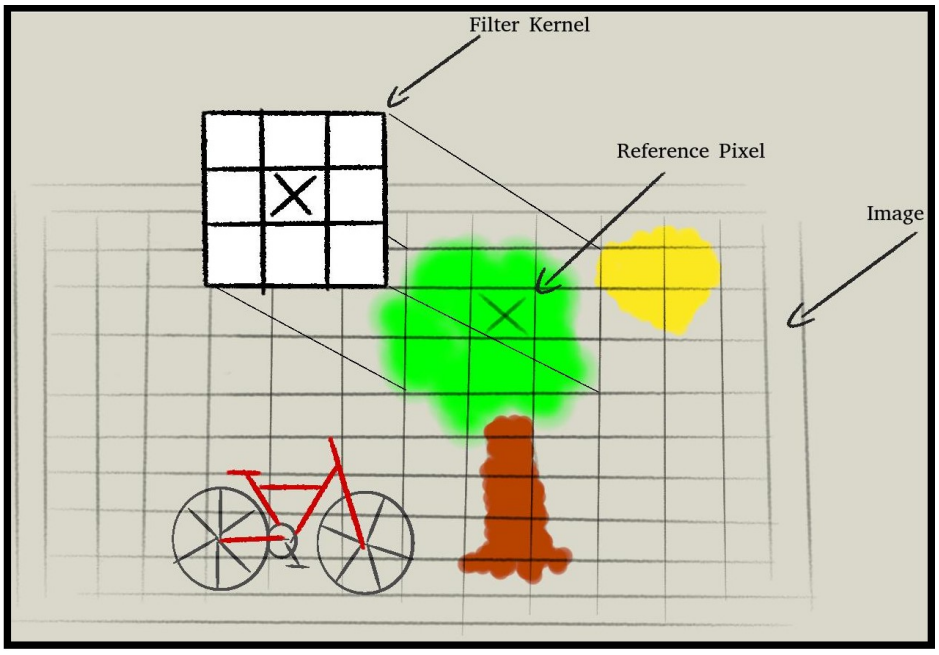
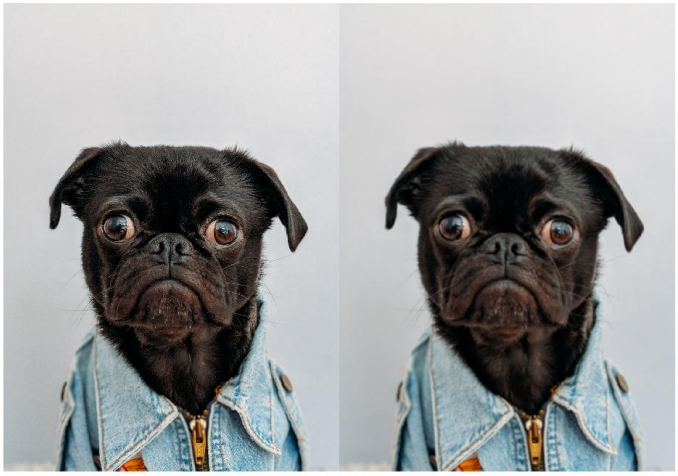


Figure 1.4: Visualization of a Filter Kernel Application

A 'box filter', for example, outputs the average of its inputs because the filter weights are evenly distributed. By passing this filter over an image its sharpness is reduced giving a smoothing or blurring effect. This can be observed in Figure 1.5a.



(a) Left: Photo by Charles Deluvio. Right: Application of 16×16 box filter.

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

(b) Box Filter Kernel.

Figure 1.5: Box Filter application and kernel.

1.2.1 Correlation

The application of a linear filter $h(u, v)$ to an image $f(i, j)$ may be described as follows

$$g(i,j) = \sum_{u=-k}^{k} \sum_{v=-l}^{l} f(i+u, j+v)h(u,v) \tag{1.2}$$

$g(i, j)$ is the output image. Performing correlation with a filter may be notated more concisely by the correlation operator.

$$g = f \otimes h$$

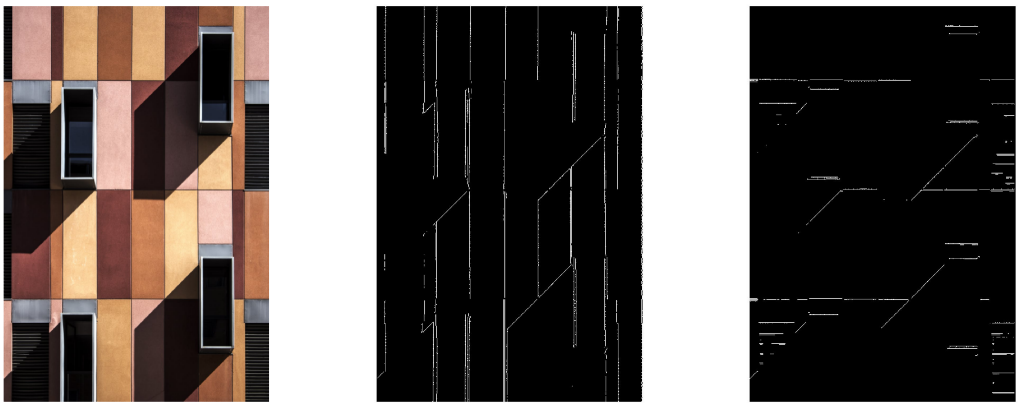
Correlation measures the similarity between two signals. Both digital images and linear filters are two dimensional signals. Performing correlation between them will yield an output image where the highest values correspond to where the image and filter were most similar [1]. This useful because if you wish to emphasise a feature in an image it can be done by correlating it with a filter that describes that feature. For example, if you wished to exagerate vertical and horizontal lines you could use Sobel filters as in 1.6.

$$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix} \qquad \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$$

(a) Horizontal Sobel Filter Mask (b) Vertical Sobel Filter Mask

Figure 1.6: Sobel Filters

The use of negative weightings means that values next to an edge are diminished and the positively weighted line sections (the 2s) strengthen line features. Notice in figure 1.7 how the lines have a high values (white) and regions that aren't lines are low valued (black).



(a) Image by Simone Hutsch (b) Vertical Sobel Filter (c) Horizontal Sobel Filter

Figure 1.7: Application of Sobel filters to exagerate lines.

Correlation is *shift invariant*, which means that it does the same thing no matter where in an image it is applied. To satisfy this property correlation may be superpositioned

$$a(f_1 + f_2) = af_1 + af_2$$

and abides by the shift invariance principle

$$g(i,j) = f(i+k, j+l) \Leftrightarrow \ (h \circ g)(i,j) = (h \circ f)(i+k, j+l)$$

Correlation has the side effect of flipping both horizontally and vertically the location of output points relative to location the center point (*reference point*) in the original image which may be undesirable.

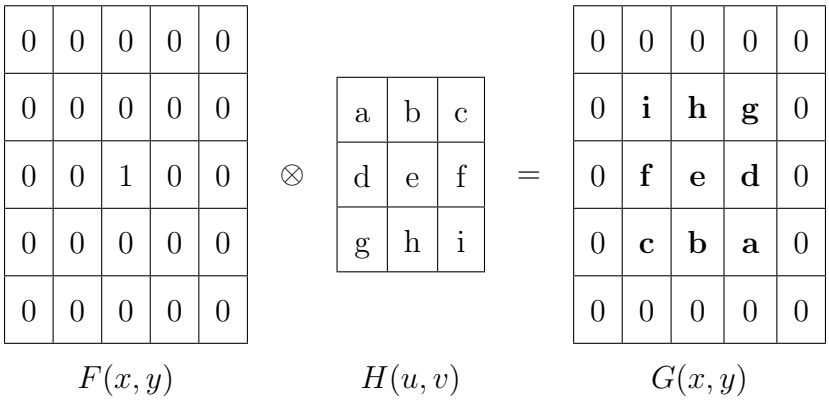


Figure 1.8: Correlation of a filter and an image.

1.2.2 Convolution

Convolution is also a linear operation that is shift invariant. It is very similar to correlation except that where correlation measure similarity between signals convolution measures the effect of one signal on another. It is described mathematically by the expression,

$$g(i,j) = \sum_{u=-k}^{k} \sum_{v=-l}^{l} f(u,v)h(i-u,j-v)$$

Notice that the filter $h(i,j)$ is rotated 180 degrees. This causes the output's orientation to match the original image. Convolution may be notated as follows,

$$g = f * h$$

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

$F(x,y)$

$*$

a	b	c
d	e	f
g	h	i

$H(u,v)$

$=$

0	0	0	0	0
0	**a**	**b**	**c**	0
0	**d**	**e**	**f**	0
0	**g**	**h**	**i**	0
0	0	0	0	0

$G(x,y)$

Figure 1.9: Convolution of a filter and an image.

1.3 The Gaussian

1.4 Clustering

Clustering is a method of segmenting an image into disjoint sets known as classes. This is useful for indentifying types of features or objects in an image. For example in Figure 1.10 the toy car has been identified out of the background.l

(a) Image by Gustavo, Upsplash.

(b) Segmentation of car from background.

Figure 1.10: Segmentation of toy cars using K-Means Clustering.

In the context of an image generally every pixel is individually classified however groups of pixels known as superpixels may also be classified. Entities are classified based on the similarity of their features. For example, a pixel's intensity is a feature that could be used to cluster it with other pixels of similar intensity. As seen in (1.3) an entity may be represented by any number of features, known also as dimensions, d.

$$\vec{v} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix} \tag{1.3}$$

1.4.1 K-Means

There are many methods with which to segment data by clustering but K-Means is very popular for its simplicity and speed. Data points can belong in any one of K clusters whose average value is a centroid. Centroids may initially be randomly selected or be members of the given data set.

Data points or samples are placed in the class of the centroid that is the closest, where the distance between them is the Euclidean Distance between their feature vectors as in

1.4.

$$D(\vec{p}, \vec{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_n - q_n)^2} \tag{1.4}$$

The algorithm that governs K-Means clustering is as follows:

1. Place the initial K centroids.

2. Assign all data points to their nearest centroid.

3. Update the centroids' values to be the mean of all data points in their cluster.

4. Repeat steps 2 and 3 until cluster centroids no longer move (converge) or until a desired number of iterations performed.

K-means benefits greatly from its low computational cost to the point where often it is performed a number of times with different initial class centroids to yield a result with the smallest variance in each cluster. It suffers, however, from the implicit assumption that cluster sizes will be approximately the same. This is because the algorithm seeks to minimize variance (spread) in each cluster hence the 'ideal' centroid placement will form distributions spherically about centroids. This method cannot disentangle overlapping samples from different classes. In Figure 1.11 approximately spherical clusters can be oberserved and the clear cutoffs between clusters ignores the class ambiguity of samples at the boundaries, this is called *hard assignment*, as opposed to a *soft assignment* that will consider the relative probabilities of a sample's class membership.

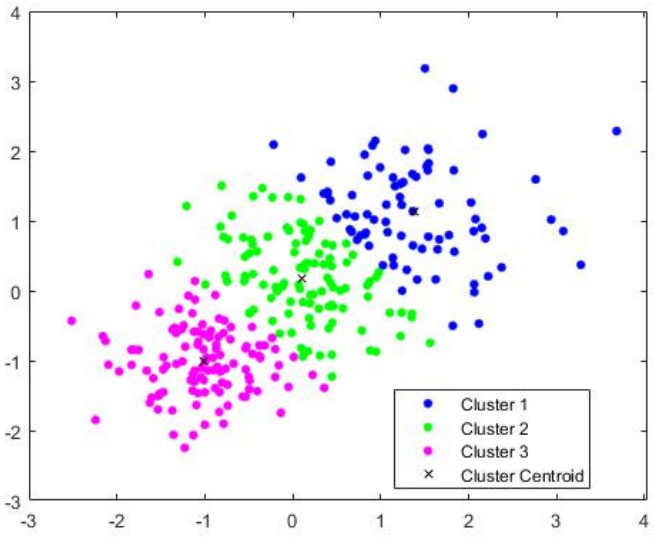
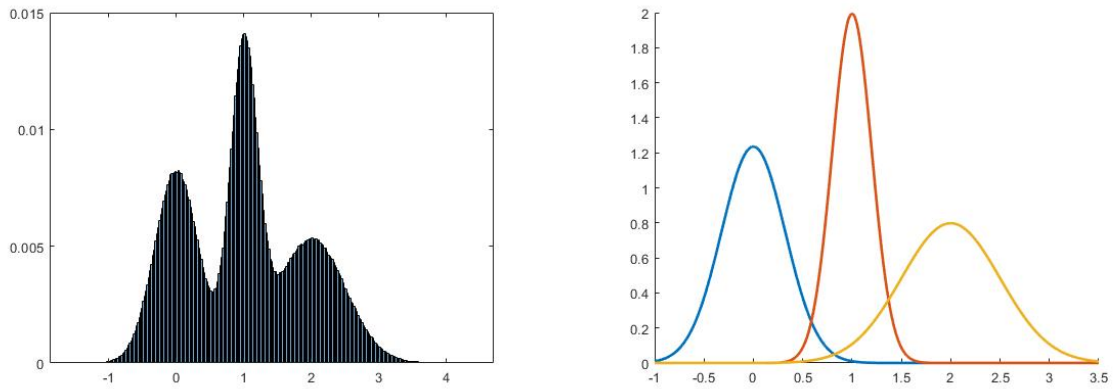


Figure 1.11: K-Means clustering performed on random data.

1.4.2 Mixtures of Gaussians

Mixtures of Gaussian is method of clustering data that's more complex than K-Means but is more effective and is able to disentangle ambiguous samples by considering probabilistic classification, known as soft assignments.

Each cluster is modelled by a Gaussian that gives the probability density of a certain values (a sample) belong to that class. The Gaussian is initially developed by taking a histogram of all values. For one dimnesional data it would appear as in Figure 1.12a. A histogram can be created for any N-dimensional data set but it becomes increasingly problematic to visualize with each additional dimension.



(a) Normalized Histogram of 1D random samples.

(b) Individual Gaussians derived from 1D histogram.

Figure 1.12: Formulation of Mixture of Gaussians.

The point of interest present in the data histogram is the domain value of the

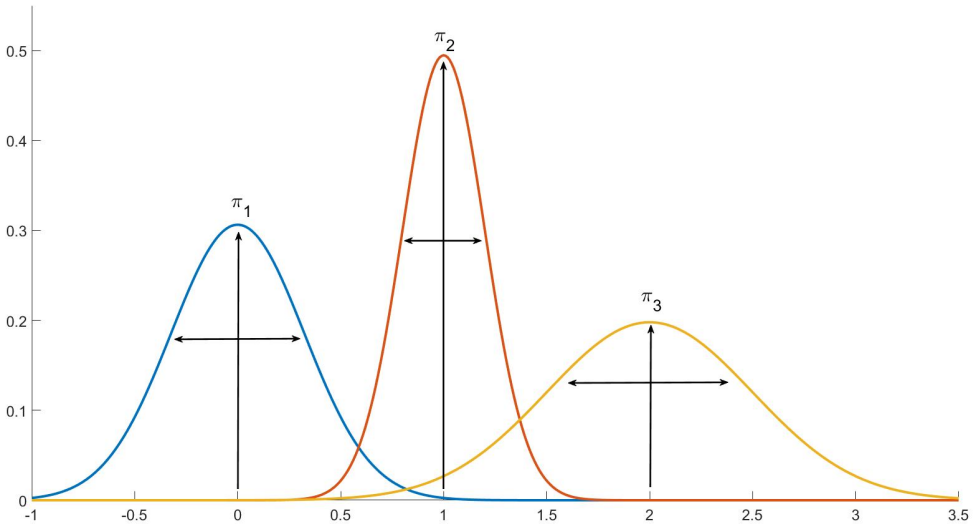


Figure 1.13: Individual Gaussians with scaled weightings.

This method is perhaps explained with an example.

Expectation Maximization

1.4.3 Background Subtraction

This is performed by identifying a background image and then subtracting this image from subsequent frames of video to determine what has changed and hence what is a foreground image.

1.4.4 Expectation Maximization

This is an optimization scheme used to fit a Gaussian Mixture Model. It is an iterative method that faurantees convergence to a local maximum in a search space.

Chapter 2

Design

The design of the system is compromised of a number of submodules.

[Insert Block Diagram of Submodules Comprising Whole System]

[Then Breakdown Each Submodule]

[Why was a CNN not used?]

Bibliography

[1] Sabyasachi Sahoo. Deciding optimal kernel size for cnn.