Smartphone-Camera Model Identification Using Digital Images

AKUA GYANBA BINEY
APPLIED COMPUTING DEPARTMENT

A THESIS SUBMITTED FOR THE DEGREE OF
MASTER OF SCIENCE IN INNOVATIVE COMPUTING
TO THE SCHOOL OF SCIENCE AND MEDICINE AT THE UNIVERSITY OF BUCKINGHAM

DECEMBER 2012

ABSTRACT

Smartphone-camera model identification using digital images analyses the relationship between modern smartphone cameras and their related images. These discovered relationship and features are used to develop a prototype software model that would be able to predict originating cell phone devices of images presented to the model. The main image processing tool used in throughout the project is Matlab. A total of 64 features broadly categorized into colour features, wavelet features and image quality features are extracted from all 1800 images captured with 10 different smartphone devices. The project tries to analyse the image features to find out which is more influential and effective in accurately identifying source devices of images under different scenarios especially in cases where images have undergone some sort of editing. A model(s) is trained with these features to identify smartphone source devices of images. A binary class turned to multiclass support vector machine (SVM) is the classifier used. Though the project focuses on phone models (brands), research also compared similar models from the same manufacturer. This project finds wavelet and image quality features highly effective in distinguishing source devices of images from different manufacturers as well as models from the same manufacturer. Results from the experiments are good and promising in correctly identifying source smartphone devices.

Keywords – Image feature, Colour features, Wavelet, Image quality metrics, Smartphone Camera Model Identification, Support vector machine, forensics

ACKNOWLEDGEMENTS

Almighty God

For the gift of life, abundant wisdom, strength and direction he provided at all times

My Supervisor

Dr. Harin Sallehewa *who tirelessly and promptly provided me with all the possible support needed for this project, most importantly allowing me much freedom to explore and define my ideas for this project.*

Lecturers and staff of Applied Computing Department

Deepest gratitude to lecturers in the Applied Computing department who taught me courses that enable me bring to fruition, this project.

Also, appreciation goes to administrators of the department for their prompt response to issues.

My Parents

Sincere thanks to my mum for her unending support in prayers and my dad who wants in return for his support, to dedicate this project to him.

My Friends

Big thanks to all my friends (both old and new) for their encouragements, supports through the changing scenes of my stay in Buckingham University.

DECLARATION

I declare that this paper, hereby submitted for the degree of Masters of Science (Innovative Computing) at Department of Applied Computing, School of Science and Medicine, University of Buckingham has not been submitted earlier on by me or anyone else for a degree at this or any other university. That, it is my own work and materials consulted have been properly referenced.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

The rising need to prove the originality and authenticity of digital images used in court as proof of evidence and trace crimes committed using digital images such as child pornography, blackmail among others, have led to various research and studies in ways of identifying source devices of digital images. A typical scenario is copyright issues relating to ownership of an image. A case like this gets to court and the best solution is to actually trace the source of the image. Many identification methods and approaches ranging from extracting information from the JPEG or EXIF header, image histogram based methods, JPEG compression methods, and image feature based methods among others have been proposed to identify source devices of digital images. Previous research focuses more on images from typical digital cameras, printers, scanners and other computer aided or generated images. However, not so much has been done with respect to cameras embedded in smart phones not to talk about the sophisticated modern phones now available. Most images nowadays, are captured using smart phones because it is highly portable and mobile. Statistics have it that more people own mobile phones than digital cameras with the purchase of smart phones still rising high. The overall aim of the project is to identify which smartphone camera model was used to capture a given image using tools in digital image processing and analysis. The EXIF or JPEG header which contains all vital information of the image like camera device model used to take the image, image size, date the image was taken, image resolution etc. is the most assured and accurate way to find the camera that took an image in question, all things being equal. Nonetheless, current and sophisticated digital image processing tools like Photoshop, GIMP, Fireworks, Inkscape and Pixelmator can be used to edit or remove this information making it not so reliable nowadays in terms of authenticity. This project uses the Image feature-based approach. This approach analyses the relationship between a source device and it associated image. It believes that all cameras (typical digital cameras and smartphone cameras) follow the same image acquisition process so the components and configuration settings of these devices which differs from manufacturer to manufacturer and actually is a trade secret, could be a good approach to use to distinguish one device from the other. The different components and configuration settings used have significant effect on the image produced; hence features from images could be used to trace their source devices. The image feature based approach has been successfully used in previous work to identify digital cameras, scanners and printer device sources.

Digital Image processing tools will be used to extract features from acquired images. Extracted features will be analysed to get distinct features associated with each device used. Results from the analysis will be used in classifying and identifying source devices. Images will be captured using smartphones from different brands, i.e. from different manufacturers and phones from the same manufacturer i.e. of the same model. Furthermore, a smartphone camera model identification software prototype to be called SmartPHid will be designed using an image processing tool like matlab.

1.1 MOTIVATION

Crimes involving mobile phones has been long recognised by the law with the increase in availability of mobile devices on the consumer market coupled the support of various activities like web browsing, email, MMS, video chat etc. However, little research or effort is being done to curb such crimes because the law needs some sort of prove of evidence before a person could be charged guilty or not guilty resulting in many offenders going scot-free for crimes committed. Also, in most cases, the device in question needs to be available for the investigation process. What happens in the absence of the device? Methods to extract mobile device data are becoming more technical, expensive and in some situations, invasive that forensic investigators need to develop their own boot loaders to bypass factory settings. In the not too distant future, most computer application ranging from electronics, software and hardware will become an essential part of forensics which will also be gaining more popularity in the field of criminology. The motivation is to research and use existing state of the art technologies to further help improve work being done in this field by reducing runtime complexity of algorithms, reduce time for image analysis especially cell forensic image applications that will help curb crimes committed using images.

1.2 OBJECTIVES

Objectives are to discover:

- The essential components and functions of a typical digital camera
- How pictures are captured and digitised
- Understand techniques used in image feature extraction and classification
- Relationship between smartphone cameras and images they generate.
- Contribute to the forensic science society

1.3 PROJECT MANAGEMENT AND METHODOLOGY

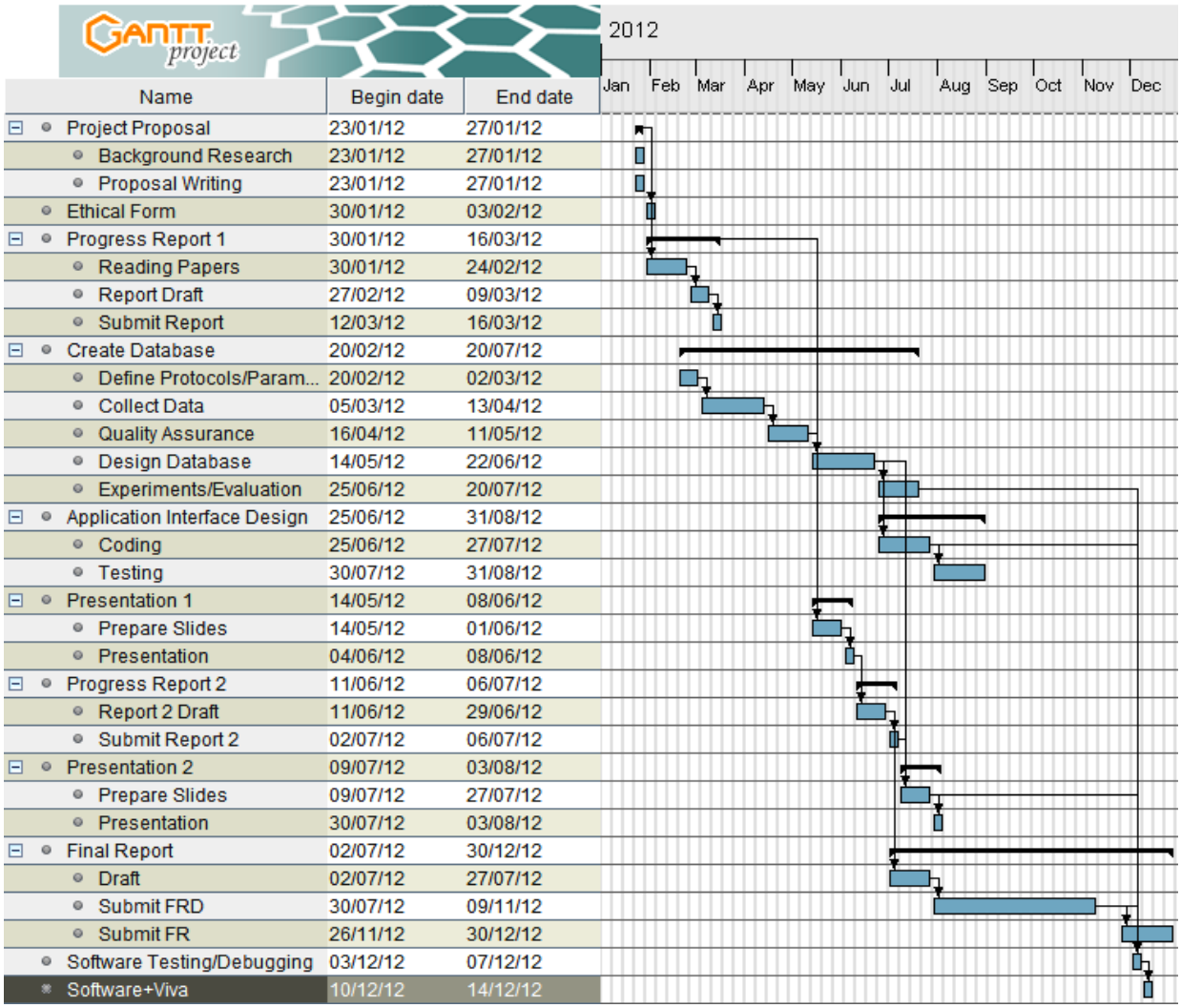
In light of the types of research; Qualitative and Quantitative, this thesis can be classified as a qualitative one because analyses focus on the quality and accuracy of performance of the identification algorithm. Another reason is that, the research is exploratory. This means result expectancy to some extent is not known. It is driven by the performance of the identification method.

To conduct the project a comprehensive literature review is done. Thereafter, the image database for the project was created using smartphone devices to take hundreds of pictures.

Next, various experiments are conducted to analyse image features and to identify best protocols and parameters to develop the prototype software. The software is called SmartPHid (Smartphone Identification v1.0).

The software development approach used in the project to develop the software is the incremental Software Development Life Cycle (SDLC) model. A partial implementation of the complete system is constructed. Other functionalities are subsequently added to finally produce the complete system.

The whole project was estimated to take approximately 40 weeks to complete. A detailed plan is presented in the chart, Figure 1 below for better visualisation.



Name	Begin date	End date
Project Proposal	23/01/12	27/01/12
Background Research	23/01/12	27/01/12
Proposal Writing	23/01/12	27/01/12
Ethical Form	30/01/12	03/02/12
Progress Report 1	30/01/12	16/03/12
Reading Papers	30/01/12	24/02/12
Report Draft	27/02/12	09/03/12
Submit Report	12/03/12	16/03/12
Create Database	20/02/12	20/07/12
Define Protocols/Param...	20/02/12	02/03/12
Collect Data	05/03/12	13/04/12
Quality Assurance	16/04/12	11/05/12
Design Database	14/05/12	22/06/12
Experiments/Evaluation	25/06/12	20/07/12
Application Interface Design	25/06/12	31/08/12
Coding	25/06/12	27/07/12
Testing	30/07/12	31/08/12
Presentation 1	14/05/12	08/06/12
Prepare Slides	14/05/12	01/06/12
Presentation	04/06/12	08/06/12
Progress Report 2	11/06/12	06/07/12
Report 2 Draft	11/06/12	29/06/12
Submit Report 2	02/07/12	06/07/12
Presentation 2	09/07/12	03/08/12
Prepare Slides	09/07/12	27/07/12
Presentation	30/07/12	03/08/12
Final Report	02/07/12	30/12/12
Draft	02/07/12	27/07/12
Submit FRD	30/07/12	09/11/12
Submit FR	26/11/12	30/12/12
Software Testing/Debugging	03/12/12	07/12/12
Software+Viva	10/12/12	14/12/12

Figure 1: Gantt chart of Project Plan

1.4 DISSERTATION OVERVIEW

This rest of the dissertation is organised as follows;

The **Literature Review** Chapter describes the image acquisition process of a digital camera and discusses some existing source image identification methods. Image features used in this project for the classification process is also discussed.

The **Image Database** Chapter describes images and devices used for the project.

The **Experiments and Results** Chapter discusses various experiments conducted in this project and the results of these experiments. Challenges of image feature selection and issues

relating to Support Vector Machines (SVM) are also discussed in this chapter. Basing on the outcome of the experiments, a software model (SmartPHid) is developed.

In the **Evaluation and Analysis** Chapter, experimental results are further analysed and compared with results from previous work.

The **Smartphone Identification Software** Chapter describes the developed software, how the system works and shows snapshots of the software model.

The last chapter covers the general conclusion, limitations and future work.

2 LITERATURE REVIEW

This section discusses the processes involved in the formation a digital image by a camera. It also discusses, image features used in classifying source devices, various identification methods and their performance in accurately identifying originating devices.

Research has been made to find out the essential components of a digital camera and how it acquires images. There are five basic components of a typical digital camera. They are optical system (lens), filters (Colour Filter Array), image sensor, image processor and storage. The image sensor which is a solid state device is the major difference between digital cameras and traditional films. Two dominant images sensors are CMOS sensor and Coupled-Charged Device (CCD). A third type of sensor which is not yet popular is the Foveon X3 sensor. Four common types of colour filter arrays are Bayer filter, RGBE (Red, Green, Blue, and Emerald) filter, CYGM (Cyan, Yellow, Green and Magenta) filter and the CYYM (Cyan, Yellow, Yellow and Magenta) filter. Configuration of the Colour filter Array also known as the Colour filter Mosaic remains practically a trade secret to the manufacturers, hence little is known about that.

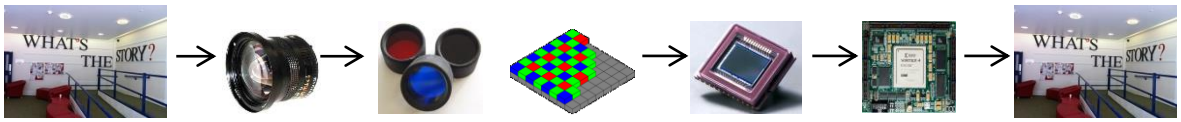
2.1 IMAGE ACQUISITION PROCESS

The various stages in acquiring an image are similar for most cameras. The optical system (lens) gathers light form the outside world or scene. The light then passes through colour filters that separate the light into red, green and blue light. These filters are placed on the pixels on the image sensor to capture colour information. Colour filters are needed because the typical photo sensors or pixels detect light intensity and therefore cannot separate colour information. The colour filters, filter the light by wavelength range, such that the separate filtered intensities include information about the colour of light. For example, the Bayer filter gives information about the intensity of light in red, green, and blue (RGB) wavelength regions.

Pixels on the image sensor does not capture colour, only brightness or light intensity of the light that falls on it by accumulating an electrical charge. More light means higher charge recorded for that pixel. Scenes with brighter or high lights records high charges and low charges are recorded for capturing lights from shadows. The image sensor has millions of photosensitive diodes which captures each pixel in the image to be produced. The charge

from each pixel is measured and converted into a digital number. This series of numbers is then used to reconstruct the image to be by setting the colour and brightness of matching pixels on the screen or printed page. It is then passed to the image processor for further processing. In order for the camera to capture, interpolate, preview, compress, display and store an image, millions of calculations are performed by the image processor.

Using a process called interpolation, the camera's image processor computes the actual colour of each pixel by combining the colour it captured directly through its own filter with the other two colours captured by the pixels around it. How well it does this is affected in part by the image format, size, and compression selected. Further processing operations include demosaicking which reconstruct a full colour image from the incomplete colour sample output from an image sensor. Gamma correction adjusts an image for accurate display. White point correction separates effects from the light source from the resulting image without changing the actual content of the image and JPEG compression balances the reconstructed image quality and the compressed file size. The image is then stored.



SCENE LENS FILTERS SENSOR PROCESSOR IMAGE

Figure 2: Camera Image Processing Pipeline

2.2 IMAGE FEATURES AND IDENTIFICATION METHODS

The process of predicting source devices in general is considered as a blind process because little or no knowledge is known about configuration settings and algorithms used by different cameras in producing an image. This is the major problem when it comes to identifying an image source device, hence leaving the choice of studying critically images produced from various cameras and trying to draw some differences from its features. The different approaches to identifying source devices can be categorised according to which parts of the camera is being examined.

Two image features namely statistical features of different images and statistical features of prediction errors were proposed by [1]. The statistical features of different images are obtained by applying filters of size 3x3 and 5x5 on images to get some differences. An absolute operation and log transformation are performed on the different images acquired

from using the filters. Then, the mean and standard deviation of the log transformed absolute values is calculated. For the statistical features of prediction errors, [1] argues that strong correlation exists across a natural image, particularly, in smooth regions. Thus, pixel values in smooth regions can be predicted from their neighbouring pixels with high accuracy. Therefore the mean and Standard deviation of linear prediction errors are used as statistical features of prediction errors.

The Lateral Chromatic Aberration features were used by [2] to identify camera sources. The method was formerly proposed by [3] to detect image forgeries. These features were used later by [4] to identify low resolution cameras such as cameras in mobile phones. He described chromatic aberration as the phenomenon where light of different wavelengths fail to converge at the same position on the focal plane. Explaining the two types of chromatic aberration (longitudinal and lateral chromatic aberration), they focused on the lateral chromatic aberration features that causes different wavelengths to focus at different positions on the image sensor of a camera. Distortion parameters between the red and green channels and parameters between the blue and green channels in the images were estimated and used as image features for the classification process.

[5] examines distortions that exist in the lens of cameras. The lens radial distortion was measured and used as a fingerprint to identify source image devices. They estimated the lens radial distortion features using a straight line method that extracts distorted line segments and measures the distortion error between the distorted line segments and the corresponding straight lines.

Another method that can distinguish image devices from the same brand and model is the sensor pattern noise approach proposed by [6]. This method focuses on the flaws of the image sensor. The sensor pattern noise serves as a unique identification fingerprint for the device. This was estimated by averaging the noise of multiple images from the device using a denoising filter. The unique fingerprint is used to trace back to the originating image device.

Methods based on Colour Filter Array (CFA) interpolation processes have also been used in solving some problems in the identification of image source devices. The method assumes that the CFA interpolation process leaves similarities across adjacent bit planes in an image. [7] researches into the interpolation process to find the correlation structure existing in the different colour bands that could be used to identify image source devices. Also, [8] used a coefficient matrix obtained from a quadratic pixel correlation model to identify source

devices. The coefficient matrix was used as input to a feed-forward Back Propagation networks for the classification process.

An auto-white balance approximation method is used by [9] which uses only image quality metrics features of the re-balanced or filtered image to identify source devices. This new approach is based on intuition that the white balance operation performed by the image processor removes all irregular colour casts so re-applying a white balance would actually not change the image.

Metrics for measuring similarities between bit planes like the binary similarity measures was used by [10] in addition to image features like image quality measures and higher order wavelet statistics to identify camera devices.

This project employs a set of image features studied by [11] for classifying originating source devices. These features are broadly categorized into Colour Features, Wavelet Domain Statistics features and Image Quality Metrics Features and are elaborated in the following sections.

2.2.1 COLOUR FEATURES

Colour processing, demosaicing and Colour Filter Array (CFA) configuration have significant effect on any image produced by a digital camera. There are about four types of Colour Filter Arrays used by digital camera manufacturers. They are the Bayer (RGB) filter, RGBE (Red, Green, Blue, and Emerald) filter, CYGM (Cyan, Yellow, Green and Magenta) filter and the CYYM (Cyan, Yellow, Yellow and Magenta).

Configuration of the CFA, fine tuning of camera components, white point correction algorithm and demosaicing or CFA interpolation algorithm for each camera remains a commercial secret to manufactures. However, some differences in the colour or colour information for different camera models could be computed with specific features. These features (colour features) according to [11] could be used to in addition to other features to determine source devices of a digital image. The colour features describes the colour reproduction of a camera. These features according to [11] can be used to distinguish a camera model from another based on the blend of demosaicing algorithm, CFA configuration and white point correction.

The colour features include:

- Average Pixel Value: This measures the average (mean) values of the three RGB channels of an image.

- RGB Pairs Correlation captures the variance in relationship of RG, RB, and GB colour channels
- Neighbour Distribution Center of Mass calculates the number of pixel neighbors for each pixel value. Distribution gives an indication of the sensitivity of a camera to different intensity levels. combination
- RGB Pairs Energy Ratio. This is used in white point correction. They are:
    - $E1 = |G|^2 / |B|^2$
    - $E2 = |G|^2 / |R|^2$
    - $E3 = |B|^2 / |R|^2$

2.2.2 WAVELET DOMAIN STATISTICS FEATURES

Images can be represented both is the spatial and frequency domains. Wavelets features are also instrumental in areas of image compression, image coding, noise removal and texture synthesis. Previous studies in the field of steganalysis have proved wavelet statics features to be of high importance in detecting hidden messages in an image [12]. He found that the decomposition of an image exhibits higher order statistical regularities that is highly affected when the image is altered though it is imperceptible to the human eye. Wavelet features describes the relationship between the sub bands. In other words, how closely the sub band coefficients co-vary from each other. Features are obtained by transforming an image into the frequency domain and then decomposing each colour band into sub bands using a biorthogonal 9/7 wavelet filter, e.g. Haar wavelet. Then the mean, variance, skewness and kurtosis of the 3 resulting high frequency sub bands are computed forming the ideal wavelet statistics features [1]. These statistics characterize the basic coefficient distributions. These features were borrowed by [11] in addition to other features for source device identification of test images. Also wavelet features could be computed by taking the average value of coefficient in each colour channel and each sub band. Convert the image into its grayscale version and compute the standard deviation and skewness of each sub band [2].

2.2.3 IMAGE QUALITY METRICS FEATURES (IQM)

The image quality metrics features measures noise and the sharpness or quality of scene reproduction by the lens of a camera. It offers more detailed differences between an image and its distorted version. Good image quality measures, well reflects the distortion on an image resulting from, compression, blurring, noise, sensor inadequacy etc. Hence, these measures contribute a lot when it comes to feature extraction processes that are used in prediction based algorithms. IQM features according to literature can be categorized into six based on the type of information they are using [13].

They are: Pixel difference-based measures, Correlation-based measures, Edge-based measures, Spectral distance-based measures, Context-based measures and Human Visual System-based measures. For the purpose of Image Source device identification, the Pixel difference-based measures, Correlation-based measures and the Spectral distance-based measures are considered.

C and $\tilde{C}$ represent the original image and its denoised version, respectively. (i, j) and (u, v) are the coordinates of an image pixel in spatial and transform domain respectively. N X N is the image size. K(=3) refers to the three colour bands. γ = 1, 2, $\Gamma_k(u, v)$ is the Discrete Fourier Transform (DFT) of the kth band image. $M(u, v) = |\Gamma(u, v)|$. $\varphi(u, v) = \arctan(\Gamma(u, v))$.

$\lambda = 2.5 \times 10^{-5}$. Γ

2.2.3.1 Pixel Difference-Based Measures.

These measures calculate the difference between two images i.e. original image and its distorted version based on their pixel differences. Pixel Difference Based Measures include:

- Minkowski Difference also known as Maximum Difference

$$MD = \max_{i,j} \left\| C(i,j) - \tilde{C}(i,j) \right\|$$

- Mean Absolute Error and RMSE

  Mean absolute error with γ = 1 and γ = 2 gives the Rooted Mean Square Error

$$MAE = \frac{1}{N^2} \left[\sum_{i,j=0}^{N-1} \{ \frac{1}{K} \sum_{k=1}^{K} | C_k(i,j) - \tilde{C}_k(i,j) | \}^{\gamma} \right]^{1/\gamma}$$

- Mean Square Error

$$MSE = \frac{1}{K}\frac{1}{N^2}\sum_{i,j=0}^{N-1}\left\| C(i,j) - C(\tilde{i},j)\right\|^2$$

- Normalize Absolute Error

$$NAE = \frac{1}{K}\sum_{k=1}^{K}\frac{\sum_{i,j=1}^{N}\left|C_k(i,j) - \tilde{C}_k(i,j)\right|}{\sum_{i,j=1}^{N} C_k(i,j)}$$

- Normalize Mean Square Error

$$NMSE = \frac{M * N * MSE}{\sum_{i=1}^{M}\sum_{j=1}^{N} C^2(i,j)}$$

- Peak Signal to Noise Ratio

$$PSNR = 10 * \log_{10}\left(\frac{R^2}{MSE}\right)$$

2.2.3.2 Correlation-Based Measures

This measure, unlike pixel difference-based measures calculates the similarity between the two images. Therefore, it can happen that minimizing difference-based measures equals maximising correlation-based measures. Similarity between two images can be calculated in terms of correlation function. These functions include:

- Structural Content

$$SC = \frac{1}{K}\sum_{k=1}^{K}\frac{\sum_{i,j=1}^{N} C_k(i,j)^2}{\sum_{i,j=1}^{N}\tilde{C}_k(i,j)^2}$$

- Normalized Cross Correlation

$$NCC = \frac{1}{K}\sum_{k=1}^{K}\frac{\sum_{i,j=1}^{N} C_k(i,j)\tilde{C}_k(i,j)}{\sum_{i,j=1}^{N} C_k(i,j)^2}$$

- Czekanowski Correlation

$$C3 = \frac{1}{N^2}\sum_{i,j=1}^{N}\left\{1 - \frac{2\sum_{k=1}^{K}\min\left(C_k(i,j), \tilde{C}_k(i,j)\right)}{\sum_{k=1}^{K}\left(C_k(i,j) + \tilde{C}_k(i,j)\right)}\right\}$$

2.2.3.3 Spectral Distance-Based Measures

These measures calculate the difference from the Fourier Transform of two images. The spectral distance based measures are;

- Spectral Magnitude Error

$$SME = \frac{1}{N^2}\sum_{u,v=1}^{N}|M(u,v) - \tilde{M}(u,v)|^2$$

- Spectral Phase Error

$$SPE = \frac{1}{N^2}\sum_{u,v=1}^{N}|\varphi(u,v) - \tilde{\varphi}(u,v)|^2$$

- Spectral Phase Magnitude Error

$$SPME = \frac{1}{N^2}\left(\lambda\sum_{u,v=1}^{N}|\varphi(u,v) - \tilde{\varphi}(u,v)|^2 + (1-\lambda)\sum_{u,v=1}^{N}|M(u,v) - \tilde{M}(u,v)|^2\right)$$

- Block Spectral Magnitude Error

$$BSME = \underset{l}{median}\, J_M^l \quad \text{where} \quad J_M^l = \frac{1}{K}\sum_{k=1}^{K}\{\sum_{u,v=0}^{b-1}\left(\left|\Gamma_k^l(u,v)\right| - \left|\tilde{\Gamma}_k^l(u,v)\right|\right)^{\gamma}\}^{1/\gamma}$$

- Block Spectral Phase Error

$$BSPE = \underset{l}{median}\, J_\phi^l \quad \text{where} \quad J_\phi^l = \frac{1}{K}\sum_{k=1}^{K}\{\sum_{u,v=0}^{b-1}\left(\left|\phi_k^l(u,v)\right| - \left|\tilde{\phi}_k^l(u,v)\right|\right)^{\gamma}\}^{1/\gamma}$$

- Block Spectral Phase Magnitude Error

$$BSPME = \underset{l}{median}\, J^l \quad \text{where} \quad J^l = \lambda J_M^l + (1-\lambda) J_\phi^l$$

2.3 COMPARISON OF THE DIFFERENT METHODS

Experiments on original and constant size (scaled or cropped to the same size) images conducted by [2] found some features not relevant in camera source identification. The RGB Pairs Correlation feature was found to be irrelevant for original size images but more important for constant size images. The Lateral Chromatic Aberration features appeared irrelevant for camera source identification because those features depend on the focal length which is fixed for low resolution cameras but not for digital cameras. The RGB Pairs Energy Ratio features were found not relevant [2]. Colour features based on gray world assumption was also found not relevant because most cameras use refined white-point corrections. But this was not the case in [9] experiment on cameras of different models, where all white balancing methods were based on gray world assumption and the results obtained were of higher accuracy. On the other hand, all image quality metrics features and the wavelet statistics features according to [2] were important.

[14] conducted a critical test to investigate the accuracy and reliability of the feature based camera model identification method. They stated that the number of images used in training each device improves identification accuracy. A total of 9487 images from 44 cameras were used in the test as compared to 150 images from 4 cameras used by [15], 5 cameras used by [11] and 13 cell phones used by [10]. Results from the test by [14] illustrates that increasing the number of images used for training from 10% of all images to 90% of all images increased accuracy from 74.7% to 95.65% respectively. [14] also reports that image size had a decreasing effect on the performance from 95.65% to 89.66%. This effect was testified in an experiment carried out by [2] which tested for robustness of the feature based identification method using transformed images. Original size images gave an accuracy of 93.56% while constant size images and cropped images gave an accuracy of 86.79% and 85.46% respectively. Works from [15] and [14] agrees that scene illumination (outdoor and indoor scenes) affects image formation and hence, the accuracy.

[14] further states that the dependence between scene content and calculated features is negligible therefore capturing scenes with similar content are really not important. Also it stated that mixing image set scenes that covers different camera settings produced reliable results.

Comparing experiments conducted on the identification of different camera brands by [9], and the naïve test by [14], results showed an overall correct model identification performance of 97.79% for [14], whereas [9] performance was 99.26%. Likewise an experiment by [9] on cameras of different models from the same manufacturer had a high prediction rate of 98.61%.

The major challenge in all the camera model identification methods is to distinguish among cameras of the same model (same manufacturer and same model). An intra-camera experiment conducted by [14] on 4 devices of Canon Ixus camera model series had overall correct identification of 79.67%. With 5 cameras of Casio EX-Z150 overall camera identification was 18.64% and with two Nikon D200 SLR cameras identification was 98.41%. [14] states that the reason why the Nikon D200 SLR camera had a higher correction identification maybe attributed to the fact that the model deviation of professional cameras are higher than normal consumer cameras.

This however improved in experiments carried out by [9]. 5 cameras of Casio EX-Z150 had overall accuracy of 98.57%, 5 cameras of Kodak M1063 gave an average accuracy of 98.47% and Nikon CoolPixS710 gave an average accuracy of 98.78% clearly showing the relevancy of image quality features in distinguishing accurately among cameras of the same model. The white balance approximation method also scales well for an increasing number of different cameras.

To further test for the robustness of the method proposed by [9], images used in the experiments were exposed to three most common manipulations which are Double JPEG compression, Noise and Resizing. Most digital cameras store images using JPEG compression, so when an image is edited and saved, another JPEG compression occurs resulting in double compression. To further test the model, original images were compressed with 75% quality metric. Results showed a decrease in average accuracy from 99.14% of original images to 89.90% proving that double JPEG compression disturbs image quality features and accuracy gets worse with increasing number of cameras especially with cameras of close models. Noise can be introduced in an image through sensor dust, dead pixels etc. Camera noise tends to increase with time. The experiment against noise tests if an image taken some years back could still be identified in the same class as those taken recently. Therefore, they added Gaussian noise with mean zero and variance 0.01 to the original images.

Results showed an accuracy of 97.79% proving the auto-white balance approximation method's resistance to Gaussian noise and thus images taken some years back could still be identified in the same class as those taken recently. Resize images were also tested and overall accuracy was 98.59%.

2.4 SUMMARY

Identification of camera models or source imaging devices can be thought of as a process of steganaylsis which believes that image statistics captured by different cameras are different. The different image identification methods and features all have their strengths and limitations depending on the focus of the research. Some methods are best suited for high resolution image devices e.g. professional cameras whiles others work well with low resolution devices. Some also, take advantage of certain parts of a camera and such methods might fail when it comes to differentiating images based on the other parts of a camera which it was not designed for. Practically, there are no differences between cameras embedded in mobile phone devices and typical digital cameras or any other camera. Technically, the difference lies in the discriminate quality of features and their types. For example, focal length in most mobile phones cameras is fixed but varies in digital cameras. Also, Resolution is generally high for digital cameras but low for mobile phone cameras. Hence, this project uses features which cover most aspects of the image acquisition process at the same time not biased or in favour of certain components of the camera. Previous studies show that image quality features and the wavelet statistics features which can detect the slightest difference between two images in the frequency domain are more important in any camera source identification method. However, the low-level Colour features cannot be ruled out in the case of low resolution devices like smartphones. These three image features are therefore going be extracted from images and used in the identification of the smartphone camera model.

The next chapter describes the image database used for the project.

3 IMAGE DATABASE

This section describes the smartphone devices and images used that constitutes the image database used for this project. The different images sizes are listed in the section and sample images from different devices in the database are displayed.

The main novelty of this project lies with the image database. Images in the database were taken using modern smartphone devices. Most reviewed works used digital cameras. The image database for this project comprises of 600 original size images of buildings, skies, roads, fields, rooms, indoor scenes, outdoor scenes, natural environments etc. taken with 10 smartphones; 60 images from each device. A quality assurance phase was carried out to get rid of images that were distorted. Naming convention used for the images follows the format of "device_img(Number)", i.e. b_img1 is the first blackberry image in the database. In as much as it was possible, the same scenes were captured with all available devices. This actually is not supposed to affect the performance of the classifier because irrespective of the scene or environment in which the image was taken, the model should be able to correctly predict it source device. In addition, all images were cropped and resized making a total of 1800 images in the database. The project focuses on phone models; hence little attention is given to a specific or exact smartphone device. Phone models under consideration include IPhones, Blackberry phones, Samsung phones, HTC phones and Nokia phones with 1 up to 3 available devices for each brand. The 60 original size images were captured using each device with variations in camera settings, more importantly using the maximum available quality settings and resolution. Actual devices or phone models used are listed in Table 1 below. Images were scaled using Microsoft office 2010 with the document-small option. Also, for each of the original size images, a region was randomly cropped forming the set of cropped images. Details of the various resolution or sizes of images from the different devices are shown in Table 2. Sample images are shown in figure 3 and 4. Also, some images were taken from the Dresden image database for comparison purposes. Images taken from the Dresden image database are from digital cameras. They are Canon Ixus 55, Canon Ixus 70, Samsung L74, Samsung NV 15, Nikon D70 and Nikon D70s. Sample images from the different devices are shown in figure 5 - 12. In conducting experiments, some images in the database are selected at random to form the training set and the rest will be used for testing.

The percentage split of images used as a training set and testing set varies depending on the experiment.

Table 1: Smartphone devices used

SMARTPHONES	NO. OF DEVICES
BLACKBERRY-BOLD 9970	1
HTC T8788	2
IPHONE 3G	1
IPHONE 4	1
IPHONE 4G	1
IPAD 2	1
SAMSUNG ONMIA 7	1
SAMSUNG GT-S5230	1
NOKIA N8	1

Table 2: Image sizes in the database

Devices	Original Images	Scaled/Resize Images	Cropped Images
HTC	2048 x 1536	800 x 600	700 x 700
BLACKBERRY-BOLD	2592 x 1944	800 x 600	700 x 700
IPAD	2592 x 1936	800 x 598	700 x 700
IP3G	1600 x 1200	800 x 600	700 x 700
IP4	960 x 720	800 x 600	700 x 700
IP4G	2592 x 1936	800 x 598	700 x 700
SAMSUNG OMNIA 7	2560 x 1920	800 x 600	700 x 700
SAMSUNG GT-S5230	2048 x 1536	800 x 600	700 x 700
NOKIA-N8	4000 x 2248	800 x 450	700 x 700

Figure 3: Sample Original Size image from the image database

Figure 4: Sample Cropped sized images in the image database

Figure 5: Sample BLACKBERRY BOLD images

Figure 6: Sample HTC T8788 images

Figure 7: Sample IPHONE-4G images

Figure 8: Sample IPAD-2 images

Figure 9: Sample SAMSUNG OMNIA-7 images

Figure 10: Sample SAMSUNG GT S5230 images

Figure 11: Sample NOKIA N8 images

Figure 12: Sample IPHONE-3G images

3.1 SUMMARY

The image database was described in this chapter. It consists of 1800 images in total from 10 different devices: 600 original size images, 600 resized images and 600 cropped size images. Similar scenes were captured with all devices using maximum resolution settings. The section also describes the various sizes of the original and modified images. Lastly, sample images in the database from different devices were displayed in this chapter.

The next chapter discusses the experimental set up and protocols, actual experiments conducted in the project and analyses results from the experiments.

4 DESIGN OF EXPERIMENTS AND RESULTS

This section is dedicated to describing experimental setup, issues in selecting robust image features for developing an identification system and Support Vector Machines (SVM) which is the machine learning tool used in the project. The chapter also presents and analyses results from the various experiments conducted.

As mentioned earlier on, the project focuses on identifying smartphone camera models from which an image in question is more likely to originate from. The main concern here is to investigate and select which features are appropriate, robust and have significant effects in terms of accuracy in determining source devices. Feature selection process under certain situations is termed to NP-hard because a comprehensive search needs to be carried out over all possible feature subsets. Some features do have little or no influence in predicting certain image source devices. Others may have significant influence on original images but when these images undergo some form of editing, the features that were initially significant turn out to be irrelevant [16].

Feature selection for most existing source device identification systems are based on achieving high accuracy and computational efficiency of which is less sufficient in real life camera source device identification applications.

Images in real life may have undergone various images processing like resizing, masking, adjusting image brightness, red eye removal, grayscale conversion, removing and adding background etc. Hence, development of the smartphone camera model device identification application needs to take into consideration the above mentioned scenarios and have the capability to deal with such images.

The use of a classifier to accurately predict image source devices needs to be carefully studied in order to obtain accurate and best results. For this project, all experiments and investigations used the Support Vector Machine (SVM) developed by [17] in training the model(s). Support Vector Machine is defined in terms of the hyperplane and its associated linear decision function [18]. There may be more than one hyperplane that could separate positive training examples from negative training examples. SVM tries to not only to find a separating hyperplane but also an optimal hyperplane that maximises the distance between the two classes. i.e. maximum distance to the nearest example from any of the training examples.

SVMs are widely used classifiers in various fields of study or research because of its ability to generate non-linear decision boundaries using methods designed for linear classifiers. It is also flexible in modelling data from different sources, able to deal with high dimensional data and data with no fixed dimensional vector space representation [19]. However, to get high accuracy results using SVM classifiers, one needs to go beyond its obvious advantages. In training an SVM, the knowledge of pre-processing data to be modelled is very essential. Though the SVM has the capability of dealing with diverse sources of data, not carefully processing data before training an SVM classifier greatly affects the results. For example, [16] conducted an experiment on grayscale images using an informed SVM classifier and a naïve SVM classifier. Results showed 93.09% accuracy for informed SVM classification and 28.29% accuracy for naïve SVM classification.

Another concern one needs to be aware of is selecting the appropriate kernel for the classifier. Kernels are used in SVM to solve the curse of dimensionality problems by making it possible to effectively use infinite dimensions as well as deal with computational problems resulting from working with very large vectors. Kernels include linear kernel which is the simplest of all the kernels, Polynomial kernel best suited for normalized data, Gaussian kernel, sigmoid kernel and Multi-Layer Perceptron (MLP) kernel. This project uses the Polynomial and Gaussian kernel which uses the radial basis function kernel. Gaussian kernel is the most flexible kernel and it's most appropriate when little is known about the data to be modelled. Not choosing the right kernel may result in a bad kernel described by [20] as a kernel with a diagonal matrix. It has all its points orthogonal to each other. Hence, has no structure nor clusters. Too many irrelevant features could also result in a bad kernel.

Finally, parameter selection is very important and needs to be carefully chosen. Ignorant choices may result in low accuracy because these parameters have significant effect on the decision boundary. Large values selected for the soft margin constant results in high margin errors whiles small values results in a larger margin separating classes or data. Kernel parameters such as the width of a Gaussian kernel (gamma) or the degree of a polynomial kernel affects the decision boundary and determine how flexible data is fitted in an SVM. Large values for gamma parameter and degree of polynomial functions works well when a non-linear relationship exists between features. However, they are most likely to cause over fitting [19].

SVM is basically a binary classification model. In order to use it in the multi class scenario, several binary classifications will be performed and combined. There are other methods to

use a binary SVM in a multiclass environment but in terms of classification accuracy and computational cost the one-versus-one method is more useful [21]. Hence, the one-versus-one method is used. The project uses 10 devices so has 45 classifiers to train, i.e. for n number of classes n(n-1)/2 classifiers needs to be constructed where each one is trained with data from only two out of *n* classes to predict a source device. The output from each classifier in the form of a class label is obtained. Data is assigned to the class label that occur most (the class with maximum vote). A subjective decision may have to be made by the investigator or image analyst in cases where there is a tie. A common tie breaking approach is to randomly select one of the class labels that are tied. In addition to randomly selecting one of the tied classes, a decision can be made at the point where the two classes were tested. One of them will emerge the winner.

Implementing this project requires a series of experiments and tests. For all experiments, the above described features (colour, wavelets, IQM) are extracted from all images. Original size images, cropped size images, and resized images are respectively used in separate experiments. Further experiments involving mixed image sizes e.g. original size images forming the training set and cropped size images used as testing set are conducted to further test the performance of the developed model. Each image feature is computed on a different scale, so resulting feature values for the different classes are likely to have different and wide ranges, thus there was the need to bring all features to a common range. To achieve this, the extracted features are normalized before it was applied to training a model. Normalization is performed for each feature by subtracting from it the mean of all features and dividing by the standard deviation of all features. For this reason the radial basis function kernel and the polynomial kernel is used in training and predicting because they are more is more suitable for normalized data and high classification accuracy. The normalized features from the 10 devices are used to train 45 classifiers to build a model that will be used to predict other instances of the classes. To predict a class of a test image, features would be extracted from the test image. The extracted features will be normalized and presented to the different models (45 classifiers). Each of them predicts a class for the test image. Results from the models are integrated for a final decision as to which class the test image likely belongs to, in this case the originating source device. Figure 2 illustrates the described algorithm.

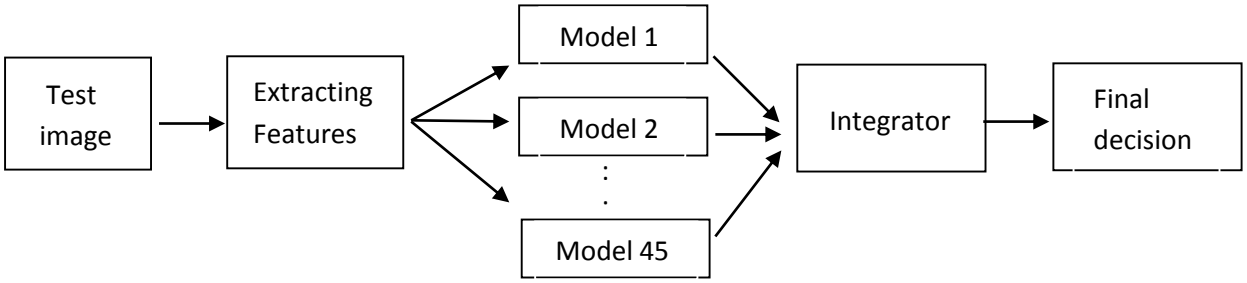


Figure 13: Algorithm for predicting smartphone source device

Three percentage split tests will be used in all experiments. In experimenting with original size images, models will be trained with 90% images from all devices and tested with the remaining 10% images. Another experiment, still using original images, 50% of the images of each device will form the training set and the remaining 50% used for testing. Then a third experiment will use 10% images as training set and 90% as testing set.

These scenarios will apply for scaled images and cropped images. Also, these experiments will be conducted for the different categories of features i.e. using only colour features, wavelets features and Image quality features. The aim of testing separately these features in different experiments is to assess which feature category is more or less useful in identifying smartphone camera models. Another experiment will combine all the features to assess the result and its performance against individual features.

Based on the outcome of the experiments and analysis of results a prototype application is developed to predict source devices of new instances.

4.1.1 PILOT TEST

Pilot tests were carried out using images from two smart phones; Samsung Omnia 7 and HTC phones. Each phone was used to capture 22 images of similar if not, the same scenes. In total 44 original (none edited) images were used for the experiments. The classifier was constructed with default values; soft margin constant c of 1, default gamma value of which equals 1/number of features and a radial basis function kernel was used for classification.

Experiment using only colour features with cost parameter =1 and gamma = 0.08 resulted in 93.18% accuracy for using all images in training and testing.

When 50% of the images from each device were used in training, i.e. the first half of images from each class, accuracy reduced to 77%. Features that showed significant effect in this category is the Energy ratios giving a result of 63% accuracy when used alone to classify test images. The mean values and the RGB correlation pairs were the next useful features giving an accuracy of 59%. Neighbour Distribution Center of Mass gave the least accuracy of 45%.

Using all wavelet features with c =1 and g = 0.08 showed 100% accuracy for training with all images of each device and testing with all images. Likewise when half of the images were used in training and the remaining used in testing 100% accuracy was still obtained. Interestingly, the mean values of the resulting high frequency sub bands of the colour channels were the most influential. It showed 100% accuracy as against 68% for variance and 50% for both skewness and kurtosis.

The mean values of all tested HTC images had positive values for the red and blue colour channels of the resulting high sub bands. Values ranged from 0.0194 to 0.1296. The green channel however, was constantly negative with values from -0.1951 to -0.1022. The opposite is said of tested images from Samsung Omnia 7 device. All images with the exception of three images had negative mean values for the red and blue high sub bands. Values ranged from-0.667 to -0.0294. The green channel had positive values from 0.0099 to 0.0849. The clear distinction of the mean values in the wavelet category led to the 100% classification accuracy for the two devices. Combining all features in this category gives 100% accuracy for training half of the images from each device and the remaining for testing.

Theoretically, mean values of sub bands or wavelet transforms is zero but with numbers like those explained above, mean values cannot be ignored or equated to zero when it comes to classification of image source devices. The mean therefore, is the most significant of all the features in the wavelet category.

Classification accuracy for using all images in training and testing showed 86.36% for all Image Quality features. Individual results are shown in table 3 below using 50% of the images in training and 50% in testing. It is important to model the behaviour of random noise in image processing applications. Gaussian noise is very useful in cases such as imaging sensors operating at low light levels [22].

Thus, the denoised version of the original image was obtained by adding Gaussian random numbers with zero mean and a unit variance to the original image and using median filter to remove the noise. Pixel difference based features were the most influential in this category.

Table 3: Image quality metrics pilot results

Features	Accuracy
All IQM features	68.18%
Pixel difference-based	68.18%
Correlation-based	45.45%
Spectral distance-based	59.09%

Tests were conducted combining all image features. An overall accuracy of 100% was recorded using all images in training and testing. Splitting the data set into 50% for training and the remaining for testing, reported accuracy was still 100%. This result was greatly influenced by the wavelet category, specifically the mean features.

Testing with images from the Dresden image database overall accuracy was 82.43%. 50% of the images were used in training and the other 50% for testing. Testing using only mean features of the wavelet category showed a result of 77%. Table 4 is a confusion matrix for the classification accuracy. Results were good in terms of inter and intra-camera similarity with the exception of Canon Ixus 55 which were mostly classified as Canon Ixus 70 and Nikon D70s.

Table 4: Results using images from the Dresden Image Database

Camera Model	1	2	3	4	5	6
Canon Ixus 55	**46.4**	20	3.2	7.2	4	19.2
Canon Ixus 70	0.92	**96.62**	0.3	1.23	0.31	0.62
Samsung L74	0.29	0.87	**93.33**	5.51	-	-
Samsung NV15	1.45	3.48	8.99	**85.50**	-	0.58
Nikon D70	3.23	5.16	-	1.29	**84.52**	5.80
Nikon D70s	-	2.1	0.53	-	8.95	**88.42**

4.1.2 EXPERIMENTS ON ORIGINAL SIZE IMAGES

All experiments in this section were carried out on original size images.

4.1.2.1 USING ONLY COLOUR FEATURES

Using only colour features, 10% of the images from each device were used in training and 90% for testing. Overall accuracy was 62% as shown in the Table 5 below. Default parameter settings were used for most experiments. BLACKBERRY-BOLD, HTC-T8788, IPAD 2, IPHONE 3G, IPHONE 4, IPHONE 4G, NOKIA-N8, SAMSUNG GT-S5230, SAMSUNG OMNIA 7 and HTC-T8788 are represented as BB, HTC, IPAD, IP3G, IP4, IP4G, NOKIA, SGT, SO7 and HTCM respectively. For clarity purposes, percentage values less than 4% are represented by an asterix (*). Also, all results are presented in a confusion matrix format. Results showed much confusion in identifying devices especially with the Nokia-N8 device. Individual highest recorded accuracy was 88.33% for IPHONE-3G and the lowest been the NOKIA-N8 device with 29.63% accuracy.

Using 50% of the images in training and the other 50% in testing, recorded accuracy was 64%. This result is slightly higher than the 62% recorded when 10% was used in training. Results are shown in table 6 below. Difference in terms of colour reproduction increased in some devices like the NOKIA-N8 whiles others decreased like IPHONE-3G. Accuracy for these two devices was 76.33% and 82.67% respectively. IPHONE-3G maintained the highest accuracy of 82.67%. IPHONE-4 recorded the lowest accuracy of 49.67% losing more than 8% to the IPAD device and 7% to IPHONE-4G device. The intra-camera similarity was a high in this case.

Using 90% of images from each device in training and the 10% in testing, overall accuracy was 76% as shown in table 7. HTCM device, which is of the same model as the HTC device, lost a great 10% to HTC. The two Samsung devices had their images misclassified to almost the same kind of devices, i.e. IPAD, IP4, IP4G, and HTCM. The overall low accuracy using colour features shows that colour reproduction by the different devices are not that different. There are more similarities in the overall colour quality than variations.

Table 5: 10% of images for training and 90% for testing

Overall Accuracy = 62%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	**59.81**	5.56	4.07	4.26	6.30	4.26	*	5.37	*	4.63
HTC	4.44	**65.93**	*	*	*	6.11	*	*	*	7.41
IPAD	5.56	4.63	**65.74**	*	*	4.44	*	*	*	*
IP3G	*	4.07	*	**88.33**	*	*	*	*	*	*
IP4	5.19	6.30	4.81	*	**69.07**	*	*	*	4.81	*
IP4G	4.26	5.00	6.67	*	6.67	**65.93**	*	*	*	*
NOKIA	8.52	8.33	9.44	6.85	9.44	9.44	**29.63**	5.74	5.74	6.85
SGT	*	5.74	7.04	*	9.07	4.26	6.30	**49.44**	7.59	5.56
SO7	*	*	5.56	*	*	5.00	5.00	*	**64.26**	5.37
HTCM	4.26	*	4.07	*	9.07	7.04	4.44	*	*	**59.81**

Table 6: 50% of images for training and 50% for testing

Overall Accuracy = 64%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	**50.67**	6.00	5.00	5.33	*	5.00	6.67	5.67	6.00	6.00
HTC	*	**50.33**	5.33	5.00	5.33	5.00	6.67	5.67	5.67	7.33
IPAD	4.33	*	**63.00**	4.00	*	6.67	*	7.33	*	5.00
IP3G	*	*	4.00	**82.67**	*	*	*	*	*	*
IP4	5.00	4.67	8.67	4.67	**49.67**	7.00	5.67	6.33	*	5.33
IP4G	*	*	5.67	5.00	*	**69.00**	*	6.33	*	*
NOKIA	4.67	*	*	*	*	*	**76.33**	5.67	*	*
SGT	*	*	4.67	*	*	*	*	**77.67**	*	*
SO7	*	*	5.33	*	6.33	5.33	4.00	7.67	**57.00**	5.67
HTCM	4.00	*	4.67	*	4.67	4.67	6.33	5.00	*	**61.67**

Table 7: 90% of images for training and 10% for testing

Overall Accuracy = 76%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	63.33	6.67	5.00	5.00	5.00	5.00	*	*	5.00	*
HTC	*	95.00	*	5.00	*	*	*	*	*	*
IPAD	*	*	76.67	*	*	*	*	*	*	6.67
IP3G	*	*	*	83.33	*	*	*	*	*	*
IP4	6.67	*	*	*	66.67	*	6.67	*	*	6.67
IP4G	*	*	6.67	*	*	83.33	*	*	*	*
NOKIA	*	*	*	*	*	*	93.33	*	*	*
SGT	*	5.00	5.00	*	5.00	5.00	*	68.33	5.00	5.00
SO7	*	*	5.00	*	5.00	5.00	*	*	80.00	5.00
HTCM	5.00	10.00	5.00	8.33	5.00	*	5.00	5.00	5.00	51.67

4.1.2.2 USING ONLY WAVELET STATISTICS FEATURES

Results from experiments conducted using wavelet features were significant like results from the pilot test. Using 10% of the images in training and the remaining 90% in testing, overall accuracy recorded was 82% as shown in table 8 with the HTC and HTCM phones having the lowest accuracy of 72.04% and 73.89% respectively. 5.19% of the images from each of these devices were classified as coming from Samsung Omnia 7 device. Revisiting wavelet results from the pilot test, the major distinction between the HTC and the Samsung Omnia 7device was the constant negative mean values for the green channel of the HTC device. Increasing the number of images from 22 to 60 would have a lot of variety where the green channel of some HTC images possibly could have positive mean value and would be misclassified. This also shows that aside the negative and positive patterns, HTC devices are similar to Samsung OMNIA 7 devices. It could also be that, they are all android phones; hence we should expect some similarities among them. Highest individual accuracy recorded was 99.63% from the IPHONE-3G device.

Overall accuracy in using 50% of the images in training and 50% in testing increased to 88% in table 9. Individual highest accuracy was 99% from IPHONE-3G and the lowest from Samsung Omnia 7 with 70.67% accuracy. Intra-camera similarity disturbed accuracy for the IPHONE 4 device which lost 5% and 6.67% to IPHONE-4G and IPAD respectively. HTCM also lost more than 5% to its HTC counterpart.

In table 10, using 90% of the images in training and 10% in testing, overall recorded accuracy was 95% with HTC and SAMSUNG GT recording 100% accuracy. Lowest accuracy was 91.67% from Samsung Omnia 7 device which lost 3% each to Samsung GT and IPHONE-4. HTCM lost its remaining 5% to the HTC device. Likewise IPHONE-4G also lost the remaining 6.67% to the IPAD device.

Table 8: 10% images for training and 90% for testing

Overall Accuracy = 82%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	95.19	*	*	*	*	*	*	*	*	*
HTC	*	72.04	4.26	*	7.22	*	*	*	5.19	5.74
IPAD	*	*	77.78	*	*	7.04	*	*	*	*
IP3G	*	*	*	99.63	*	*	*	*	*	*
IP4	*	*	5.37	*	77.22	*	*	*	*	4.81
IP4G	*	*	*	*	4.07	83.52	*	*	*	*
NOKIA	4.81	*	*	*	*	*	87.41	*	4.26	*
SGT	*	*	*	*	*	7.22	4.44	74.26	*	*
SO7	4.26	*	*	*	5.37	*	*	*	77.59	*
HTCM	*	*	*	*	*	*	*	*	5.19	73.89

Table 9: 50% images for training and 50% for testing

Overall Accuracy = 88%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	98.33	*	*	*	*	*	*	*	*	*
HTC	*	96.67	*	*	*	*	*	*	*	*
IPAD	*	*	88.00	*	*	6.00	*	*	*	*
IP3G	*	*	*	99.00	*	*	*	*	*	*
IP4	*	*	6.67	*	78.00	5.00	*	*	*	*
IP4G	*	*	6.67	*	*	85.67	*	*	*	*
NOKIA	*	*	*	*	*	*	91.00	*	*	*
SGT	*	*	*	*	*	*	*	95.33	*	*
SO7	4.67	*	5.00	*	5.00	*	4.33	2.33	70.67	4.00
HTCM	*	5.33	*	*	*	*	*	*	*	79.67

Table 10: 90% images for training and 10% for testing

Overall Accuracy = 95%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	98.33	*	*	*	*	*	*	*	*	*
HTC	*	100.00	*	*	*	*	*	*	*	*
IPAD	*	*	88.33	*	*	*	*	*	*	*
IP3G	*	*	*	93.33	*	*	*	*	*	*
IP4	*	*	*	*	95.00	*	*	*	*	*
IP4G	*	*	6.67	*	*	93.33	*	*	*	*
NOKIA	*	*	*	*	*	*	98.33	*	*	*
SGT	*	*	*	*	*	*	*	100.00	*	*
SO7	*	*	*	*	*	*	*	*	91.67	*
HTCM	*	5.00	*	*	*	*	*	*	*	95.00

4.1.2.3 USING ONLY IMAGE QUALITY METRICS FEATURES

Experimenting with image quality features also exhibited high accuracy results like the wavelet features. Actually, it was observed that it scales well with increasing the number of devices and images. 77% accuracy was attained using 10% of total images in training and 90% for testing. Accuracy increased to 88% when 50% images were used in training and 50% for testing. Test using 90% for training and 10% for testing resulted in 96% accuracy shown in table 13. This result is slightly higher than 95% reported in wavelet experiment with 90% training set.

Table 11: 10% images for training and 90% for testing

Overall Accuracy = 77%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	78.52	*	*	6.11	*	*	*	6.11	*	*
HTC	5.00	60.56	6.48	*	*	5.19	*	6.67	5.56	5.93
IPAD	*	*	88.89	*	*	*	*	*	*	*
IP3G	*	*	*	88.89	*	*	*	*	*	*
IP4	*	4.26	*	*	74.63	*	*	4.26	*	*
IP4G	5.19	*	7.78	*	*	71.48	*	*	*	*
NOKIA	*	*	*	*	*	*	94.26	*	*	*
SGT	4.07	*	*	4.07	*	*	*	79.07	*	*
SO7	4.44	5.74	4.44	*	*	5.00	5.56	5.00	63.33	4.07
HTCM	5.19	5.56	5.00	*	*	*	*	4.81	4.81	64.44

Table 12: 50% images for training and 50% for testing

Overall Accuracy = 88%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	98.00	*	*	*	*	*	*	*	*	*
HTC	*	86.67	*	*	*	*	*	*	*	4.67
IPAD	*	*	83.00	*	*	8.00	*	*	*	*
IP3G	*	*	*	90.33	*	*	*	*	*	*
IP4	*	*	*	*	90.67	*	*	*	*	*
IP4G	*	*	4.67	*	*	89.00	*	*	*	*
NOKIA	*	*	*	*	*	*	99.67	*	*	*
SGT	*	*	*	*	*	*	*	88.33	*	*
SO7	*	*	*	*	*	*	*	*	82.00	*
HTCM	4.00	6.67	4.00	*	*	4.00	4.33	*	5.67	68.00

Table 13: 90% images for training and 10% for testing

Overall Accuracy = 96%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	93.33	*	*	*	*	*	*	5.00	*	*
HTC	*	93.33	*	*	*	*	*	*	*	6.67
IPAD	*	*	98.33	*	*	*	*	*	*	*
IP3G	*	*	*	100.00	*	*	*	*	*	*
IP4	*	*	*	*	100.00	*	*	*	*	*
IP4G	*	*	6.67	*	*	90.00	*	*	*	*
NOKIA	*	*	*	*	*	*	100.00	*	*	*
SGT	*	*	*	*	*	*	*	100.00	*	*
SO7	*	*	*	*	*	*	*	*	91.67	5.00
HTCM	*	6.67	*	*	*	*	*	*	*	93.33

4.1.2.4 USING ALL FEATURES

Combining all features increased overall classification accuracy from the 96% achieved in using only image quality features to 97.5% (table 16). This accuracy was achieved using 90% images for training and 10% for testing. Using 10% images for training and 90% for testing, accuracy was 80% (table 14). In addition, 90% accuracy was achieved when 50% images were used for training and the other half for testing (table 15.). All these accuracy values although high, are quite close to the accuracy values from using wavelet and image quality features. Detailed results are shown in tables below.

Table 14: 10% images for training and 90% for testing

Overall Accuracy = 80%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	82.41	*	*	*	*	*	*	4.07	*	*
HTC	*	82.41	*	*	*	*	*	*	*	5.74
IPAD	4.81	*	84.07	*	*	5.56	*	*	*	*
IP3G	*	*	*	97.59	*	*	*	*	*	*
IP4	*	*	*	*	83.15	*	*	*	*	*
IP4G	*	*	5.00	*	*	83.70	*	*	*	*
NOKIA	4.63	*	*	*	*	*	84.63	*	*	*
SGT	*	*	*	*	*	*	6.67	75.93	4.07	*
SO7	5.00	5.93	5.37	*	*	*	*	*	65.37	*
HTCM	5.00	5.56	5.74	*	*	5.93	4.44	4.81	5.37	58.15

Table 15: 50% images for training and 50% for testing

Overall Accuracy = 90%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	92.00	*	*	*	*	*	*	*	*	*
HTC	*	86.33	*	*	*	*	*	*	*	*
IPAD	*	*	87.00	*	*	8.00	*	*	*	*
IP3G	*	*	*	99.33	*	*	*	*	*	*
IP4	*	*	*	*	89.67	*	*	*	*	*
IP4G	*	*	4.67	*	*	94.33	*	*	*	*
NOKIA	*	*	*	*	*	*	96.00	*	*	*
SGT	*	*	*	*	*	*	*	96.33	*	*
SO7	*	4.67	5.67	*	*	4.00	*	5.00	70.33	*
HTCM	*	7.00	*	*	*	*	*	*	4.00	78.00

Table 16: 90% images for training and 10% for testing

Overall Accuracy = 97.5%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	95.0	*	*	5.0	*	*	*	*	*	*
HTC	*	100.0	*	*	*	*	*	*	*	*
IPAD	*	*	91.7	*	*	8.3	*	*	*	*
IP3G	*	*	*	98.3	*	*	*	*	*	*
IP4	*	*	*	*	100.0	*	*	*	*	*
IP4G	*	*	*	*	*	96.7	*	*	*	*
NOKIA	*	*	*	*	*	*	100.0	*	*	*
SGT	*	*	*	*	*	*	*	100.0	*	*
SO7	*	*	*	*	*	*	*	6.7	93.3	*
HTCM	*	*	*	*	*	*	*	*	*	100.0

4.1.3 EXPERIMENTS ON SCALED/RESIZED IMAGES

Experiment carried out on resized images had a decreasing overall classification accuracy of 92% (see table 21 in appendix) compared to the 97.5% achieved on original size images. This attest to the fact that scaled images and modified images in general presents more challenge in image source device identification. The 92% accuracy resulted from combining all image features and using 90% images in training and the remaining 10% in testing. Image quality metrics (IQM) features were significant in this experiment with the highest accuracy of 85% according to table 22 (see appendix). Wavelet features and colour features presented nearly the same results of 77% and 75% accuracy respectively (table 23 and 24).

In using half of the images in training and the other half in testing, combining all features resulted in 73% accuracy; image quality features showed 72% accuracy; wavelet features showed 70% accuracy whiles colour features presented 65% accuracy.

Using 10% of images from each device in training and 90% in testing, combining all features presented 63% accuracy; image quality features showed 65% accuracy; wavelet features showed 60% accuracy and colour features presented an accuracy of 62%. The sharp decrease in accuracy for wavelet features can be attributed to the fact that scaled images makes the coefficient of the resulting high frequencies much closer to zero, therefore increasing similarities across devices. Wavelet features are also highly sensitive to distortions caused by resizing an image.

4.1.4 EXPRIMENTS ON CROPPED SIZE IMAGES

Similar to results reported on resized images, results on cropped images showed an overall accuracy of 93% according to table 25 (see appendix). This accuracy resulted from combining all features and using 90% of images in training. Similarly, image quality features and wavelet features were significant in this experiment reporting 85% and 84% accuracy respectively. Colour features report 71% accuracy. Removing pixels from an image affects the image quality measurements, hence making IQM features sensitive to cropping.

Using 50% of the images in training, combining all features showed 81% accuracy, wavelet features showed 78% accuracy; image quality features showed 76% accuracy whiles colour features presented 65% accuracy.

When 10% of the total images were used in training, results for combining all features, only wavelet features, only image quality features, only colour features were 68%, 71%, 68% and 61% respectively.

4.1.5 EXPERIMENTS USING ORIGINAL, CROPPED AND RESIZED IMAGES

In this sub section, the series of experiments carried out involved using the different image sizes to train and test the model. This is to further test the robustness of the model against modified images. All features were used and 90% of images of each group (image size) were used.

Using 90% of original images in training and testing with 90% cropped images, overall accuracy presented was 76.5% (table 17). Highest individual accuracy reported was 91.85% from the Samsung Omnia 7 device and the lowest being IPHONE-4 with 48.33%. Also, there were some similarities compared with previous results in terms of intra-device similarities. HTC lost 5% to its HTCM counterpart, 5.37% images from IPHONE-4G were misclassified as originating from the IPAD device and 5.56% images from Samsung GT were misclassified as Samsung Omnia 7 images.

Similar results were achieved, training with original size images and testing with resized images. Reported accuracy was 72.2% (table 18).

Table 17: Training with original images and testing with cropped images

Overall Accuracy = 76.5%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	59.07	5.00	6.30	*	*	5.74	6.30	4.26	4.81	5.00
HTC	*	79.63	4.63	*	*	*	*	*	4.44	5.00
IPAD	*	*	90.56	*	*	*	*	*	*	*
IP3G	*	*	*	89.81	*	*	*	*	*	*
IP4	*	5.74	8.70	*	48.33	8.33	*	9.44	9.26	5.00
IP4G	*	*	5.37	*	*	86.67	*	*	*	*
NOKIA	*	5.74	4.63	*	*	4.81	62.04	7.22	6.85	4.63
SGT	*	*	*	*	*	*	*	77.96	5.56	*
SO7	*	*	*	*	*	*	*	*	91.85	*
HTCM	*	*	5.37	*	*	4.81	*	*	*	79.07

Table 18: Training with original images and testing with resized images

Overall Accuracy = 72.2%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	50.74	7.22	7.41	*	*	7.04	4.63	6.11	6.85	7.59
HTC	*	82.04	5.74	*	*	*	*	*	*	*
IPAD	*	*	84.63	*	*	*	*	4.26	5.37	*
IP3G	*	4.07	*	77.96	*	*	*	*	*	*
IP4	*	5.00	7.41	1.48	45.74	7.59	5.19	9.81	9.07	5.00
IP4G	*	*	4.44	*	*	84.26	*	*	4.44	*
NOKIA	*	6.85	7.41	*	4.44	6.85	46.30	9.63	8.15	5.37
SGT	*	*	*	*	*	4.07	*	80.74	4.63	*
SO7	*	*	*	*	*	*	*	*	91.30	*
HTCM	*	*	5.00	*	*	5.74	*	*	5.19	78.15

82.2% overall accuracy was achieved when 90% cropped images were used in training and tested with 90% with original size images. IPHONE-3G reported an excellent result of 100% (table 19). Similar results were reported in table 20. 86.2% overall accuracy was achieved using resized images in training and testing with original size images

Table 19: Training with cropped images and testing with original images

Overall Accuracy = 82.2%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	60.00	*	6.11	*	5.74	4.81	7.78	9.63	5.74	*
HTC	*	91.30	*	*	*	*	*	*	*	6.67
IPAD	*	4.07	78.15	*	*	7.78	*	*	*	*
IP3G	*	*	*	100.00	*	*	*	*	*	*
IP4	*	*	5.74	0.00	67.22	5.56	4.26	7.78	6.11	*
IP4G	*	4.07	*	*	*	78.33	*	*	*	*
NOKIA	*	*	*	*	*	*	87.96	5.37	*	*
SGT	*	*	*	*	*	*	*	89.07	*	*
SO7	*	*	*	*	*	*	4.26	*	78.70	*
HTCM	*	6.85	*	*	*	*	*	*	*	91.30

Table 20: Training with resized images and testing with original images

Overall Accuracy = 86.2%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	93.89	*	*	*	*	*	*	*	*	*
HTC	*	96.48	*	*	*	*	*	*	*	*
IPAD	*	4.63	80.00	*	*	4.26	*	*	*	*
IP3G	*	*	*	99.26	*	*	*	*	*	*
IP4	*	*	7.22	*	70.19	7.04	8.52	*	*	*
IP4G	*	*	*	*	*	85.93	*	*	*	*
NOKIA	*	*	5.19	*	6.11	*	66.30	4.81	4.81	*
SGT	*	*	*	*	*	*	5.56	88.33	*	*
SO7	*	*	*	*	*	*	*	*	88.52	*
HTCM	*	5.56	*	*	*	*	*	*	*	93.15

Using cropped size images in training and testing with resize images, 77.7% overall accuracy was achieved, table 29 (see appendix). Similarly, an overall accuracy of 78.8% was reported in the experiment involving a training set of resized images and a testing set of cropped images, table 30 (see appendix).

In general, reported accuracies from the cross image size tests were not as high as the 92% achieved in the experiment involving only resize images (table 21, see appendix), 93% accuracy achieved in the experiment involving only cropped size images (table 25, see appendix) and 97.5% accuracy reported using only original size images (table 16).

4.2 SUMMARY

In this section we discussed the various smartphone devices and images that make up the image database. The image database in total is made up of 1800 images (600 original size images, 600 resized images and 600 cropped size images) from 10 camera devices. The various experiments conducted and their detailed results were also discussed. Image Quality Metrics features and wavelet features proved to be significant features in all experiments conducted.

The next section further discusses the results and compares them to results from reviewed works.

5 EVALUATION AND ANALYSIS OF RESULTS

This chapter analyses and discusses the results of the experiments conducted. Also, attempts were made to present a broad assessment of the results and also evaluate the findings with reference to results from reviewed studies.

Like results from other image source identification experiments, test results reported in this project were highly dependent on variety of factors. One of them is the image features used. On all experiments, wavelet and image quality metrics features contributed significantly to the overall accuracy recorded, especially with tests on original size images. Colour features, throughout the various experiments did not contribute much due to much similarity in terms of colour reproduction by each device. Most of the new phones like HTC, IPhones and Samsung do not really have much difference in terms of colour quality of produced image. Also another reason could be attributed to the fact that CFA configuration and demosaicing algorithm which affect the camera's colour production is not really the focus in cameras embedded in smartphones because such cameras are not meant to take high quality professional photos like their digital camera counterparts. Editing activities such as resizing and compression distorts curves, lines and edges in an image. These changes greatly affect accuracy of wavelet features because the coefficient of the high sub band of the wavelet decomposition has more to do with these edges and curves. Image quality features were found to contribute more and scores high accuracy as training data increases. Hence, it will be more useful in real life application where data set (number of devices) is more than just 10 devices. Despite its effectiveness in terms of scalability, image quality features are sensitive to editing activities like cropping which removes some pixels, thus affecting the measurement and performance of image quality metrics features.

Another observed factor affecting general performance of the model or developed classifier is the training data set. With all experiments conducted, low accuracies were recorded for using less number of images in training. Accuracy increases as the number of images used in training increases as shown in figure 14. Accuracy for using 10% of total images ranges from 60% to 80%. Using 50% of total images presents accuracy from 64% to 90% whiles using 90% of total images results in accuracies more often greater than 80%.

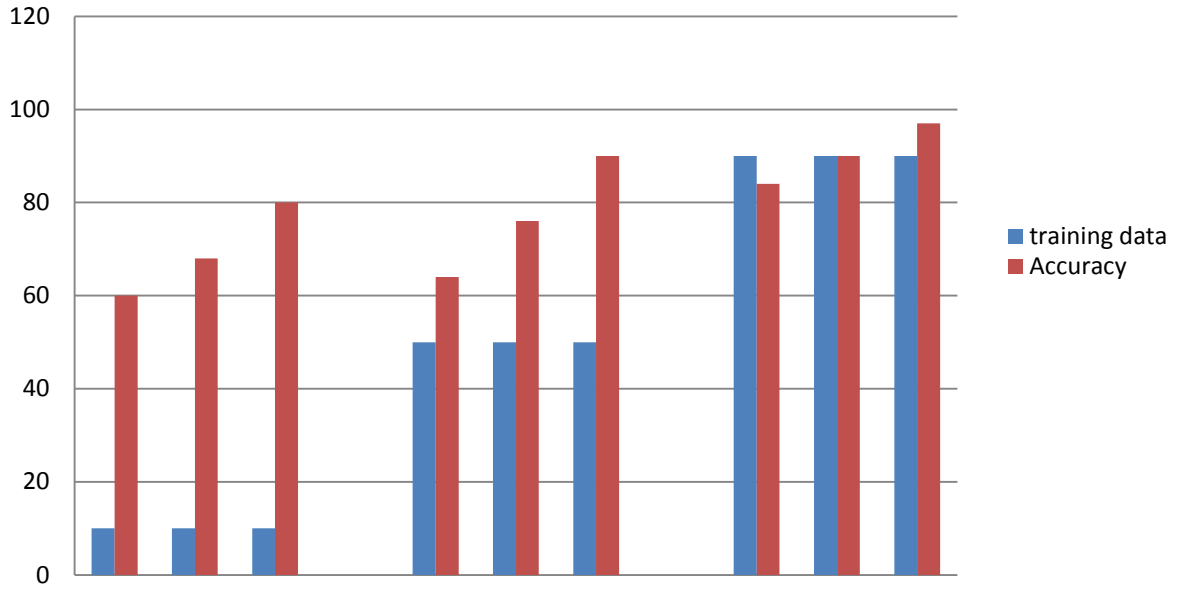


Figure 14: Results of some training set and their reported accuracy

In addition to the above observation, further calculation (hypothesis testing) was performed to help select from among the three percentage split models, a final model for the developing the prototype software [23]. Using the formulas a) and b) below by [23], the following confidence interval was calculated for the models in the various experiments: M1 = Model1 (10% training images), M2 = Model2 (50% training images) and M3 = Model3 (90% training images). The commonly used confidence coefficient of 1.96 for 95% confidence level was used.

a) $$\sigma = \sqrt{\frac{e_1(1-e_1)}{|D_1|} + \frac{e_2(1-e_2)}{|D_2|}}$$

σ is the standard deviation of the sum of errors from the different models.

b)

$$|e_1 - e_2| - Z_{\alpha/2} \cdot \sigma \leq d \leq |e_1 - e_2| + Z_{\alpha/2} \cdot \sigma$$

Where is α the percentage of confidence, e.g. 95%, $Z_{\alpha/2}$ is the confidence coefficient value from the t-distribution table at $(1-\alpha)/2$ confidence level. Therefore for 95% confidence, with an an infinity degree of freedom (df), the confidence coefficient value in the t-table is 1.96

Original size Models:

M1 = 80%, M2 = 90% and M3 = 97.5%

Models	Error Difference	lower bound	Upper bound
M1 and M2	0.1	-0.000200684	0.000222263
M1 and M3	0.17	-0.000193132	0.000229816
M2 and M3	0.07	-0.000203921	0.000219027

Scaled/Resize Models:

M1 = 63%, M2 = 73% and M3 = 92%

Models	Error Difference	lower bound	Upper bound
M1 and M2	0.1	-0.000762116	0.000844064
M1 and M3	0.29	-0.000684265	0.000921915
M2 and M3	0.19	-0.000725239	0.000880941

Cropped size Models:

M1 = 81%, M2 = 90% and M3 = 93%

Models	Error Difference	lower bound	Upper bound
M1 and M2	0.09	-0.000855734	0.000938104
M1 and M3	0.12	-0.000842005	0.000951832
M2 and M3	0.03	-0.00088319	0.000910647

From all three tables above, the difference between the models for each category of experiments according to the lower bound and upper bound values statistically are not significant. And since the speed and complexity levels of the models, were almost the same, any of the models could be selected as the final model [23]. Hence, for this project, models with the lowest error rate were selected for the purpose of comparison (accuracy) and also to develop the software prototype model.

In general, the type, number and model of devices used in the project also affected accuracy rate. Most devices used came from the same manufacturer like the IPhone devices, the HTC devices and the Samsung devices. These devices were selected purposely to test the classifier's performance on intra-camera similarity. Intra-camera similarity may lead to misclassification which affects overall performance of the classifier. However, test results according to table 16 shows good performance in classifying devices of the same model like the two HTC devices which reported 100% and devices from the same manufacturer like the IPhones with the exception of the IPhone 4G. Comparing results of the two HTC devices to that of the normal classification experiment results conducted by [4] who reported 50% accuracy for two Motorola V3i phones, it is obvious that the model developed in this project performs better in terms of intra-camera identification. He, [4] added that Lateral Chromatic aberration features alone were not enough to distinguish cameras of the same model. Most test results showed strong correlation existing between IPAD and IPhone 4G. Also, as shown in table 16 more than 6% of images from the Samsung Omnia 7 device were classified as Samsung GT-5230 device.

Making a complete comparison of results to existing works is a bit difficult due to differences in the purpose for classification, the approach or method used, the feature set, types and number of devices used, parameter settings, protocols for the experiments like the total number of images used, images sizes, image manipulation techniques and the percentage split of images used as training and testing sets. Purpose for classification maybe focused on typical digital cameras, scanners, mobile phones, specific image device or brand of the device, robust image features and so on.

However, to some extent test results in this project did not fall below expected accuracy comparing it to previous works. Reported accuracies are acceptable. Actually, there was an improvement in performance of the model/classifier especially on manipulated images. Using different image sizes in the same classification did not yield very good results. Accuracies reported ranged from 70% to 87%. For example, results on using original size images as the training set and cropped size images for testing resulted in a 76.5% overall accuracy. Accuracy rate was 86.2% when resized images were used as the training set and original images for testing.

On the other hand when the modified images are used in training, accuracy in classifying similar modified images is tends to be high. Accuracy achieved in this project was 92% and 93% as presented in table 21 and 25 (see appendix) compared to the 85.46% reported on

informed experiment on cropped images and 86.79% on constant size images by [16]. Also, the achieved 93% and 92% accuracy is comparatively better than the average accuracy of 39% and 53% reported by [1] under experiment for cropped and scaled images respectively. Overall accuracy reported on cropped image experiment by [4] was 57.76%.

Average performance resulting from the original size images experiment in this project was 97.5% which was achieved in using 90% of 600 images from 10 smartphones in training. [4] recorded an average accuracy of 92.22% using 30 images from 3 cameras. [14] also reported 97.79% overall accuracy in using 90% of 9185 images from 11 digital cameras in training. Reducing training set to 10%, achieved accuracy in this project was 77% compared to 74.05% achieved by [14] indicating that the model in this project performs fairly better in using small number of images as the training set and still maintains high accuracy as the number increases. Also overall average performance reported by [10] in using 16 phones was 95.1% as against 97.5% achieved in this project. Another analysis was made comparing resized experimental results reported by [9] which was 98.59% as against 92% achieved in this project. The big difference in accuracy can be related to the fact that same cannot be said about min and max channel values of resized images been significantly larger than values for original size images as explained by [9]. The min and max values for the colour channels were analyzed for resized and original size images in the database. It was observed that images from the IPAD, IPHONE-4, IPHONE-4G, NOKIA-N8, the two HTCs and Samsung devices had almost the same min and max values for both resized and original size images. BLACKBERRY reported same max values for all images. There were some differences in the min values but they were not significantly different. For example, the max values for some BLACKBERRY device images were 21 for both original size and it resized version whiles the min values were 23 and 27 for original size the resized image respectively. This similarity in all the smartphone devices used in this project could have resulted in the decline of accuracy (on resized images) for classifying smartphone devices whiles the high dissimilarity in digital cameras may have led to [9] achieving 98.59% accuracy. To reiterate, all these results explained, depends however, on the above mentioned reasons which makes detailed comparison difficult.

5.1 SUMMARY

Experimental results obtained in this project were further analyzed and compared to results from previous works. Accuracy rates achieved were high and acceptable in most cases. The model(s) also has the potential of accurately distinguish devices of the exact same model. The model is also robust against modified images if the training set undergoes the same or similar modification as the testing set.

Next section is devoted to the software prototype developed based on outcome of the various experiments.

6 SMARTPHONE IDENTIFICATION SOFTWARE

This chapter describes the smartphone identification (SmartPHid) software developed as part of the research. The design of the software, its usage and some snapshots of the software are presented in this chapter.

The SmartPHid software is developed to predict source devices of new images from a smartphone. The main objective of developing this software is to help facilitate an image forensic investigation process. Development of the software is based on outcome of the research conducted. Experimental protocols that resulted in high accuracy were used in developing the prototype model. 90% of total images in the database were used as the training set. Also, all features were used. Kernel parameters used were, soft margin constant c = 100, gamma = 0.016 for polynomial kernel and 0.07 for radial basis (RBF) kernel and a coef0 of 3 for the polynomial kernel.

Software development life cycle applied was the Incremental SDLC. Every single functionality added, was tested to make sure it works well with existing functionalities before moving on to add more functions.

Functional requirements of SmartPHid include uploading or opening a new image to be classified, extracting features from the new selected image, predicting source device of the image and analyzing outcome of test results. The software also allows one to save test results. The non-functional requirements are mainly the user friendly nature of the system and its error handling capability.

Figure 15 below shows the first view or main interface when the application is started. It has test options to select from and buttons on the left. Classification results will be displayed on the right side.

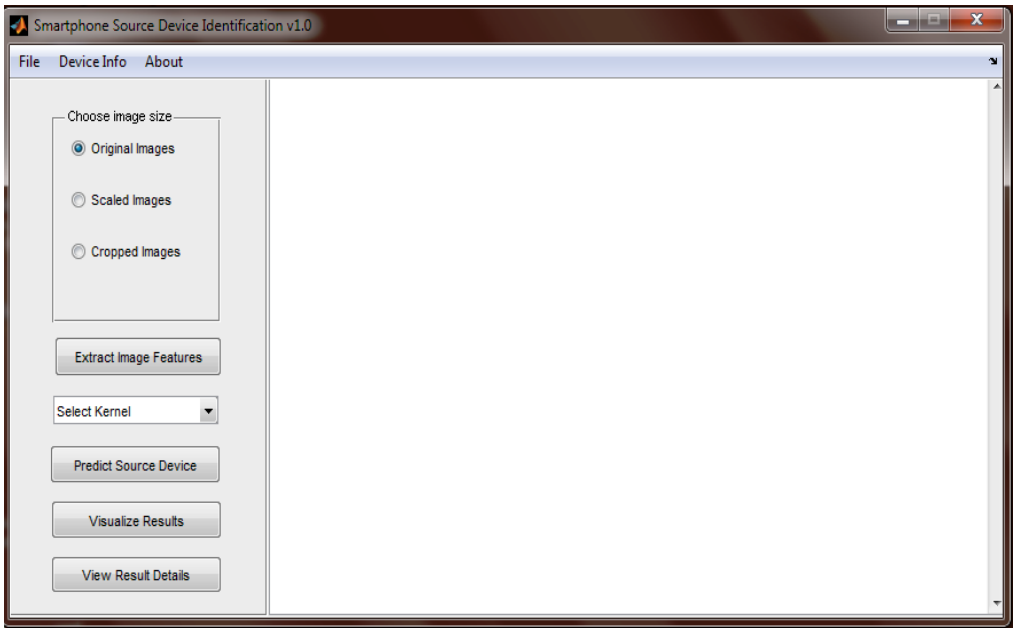


Figure 15: Main Interface

The menu bar has three items. The file menu item has 3 sub items: to select a file for testing, save test results to a text file and exit the application as shown in figure 16. An image can be selected by choosing the open sub menu item from the file menu item in figure 16 or by using the hot keys Ctrl + O to open the select file dialog box in figure 17.

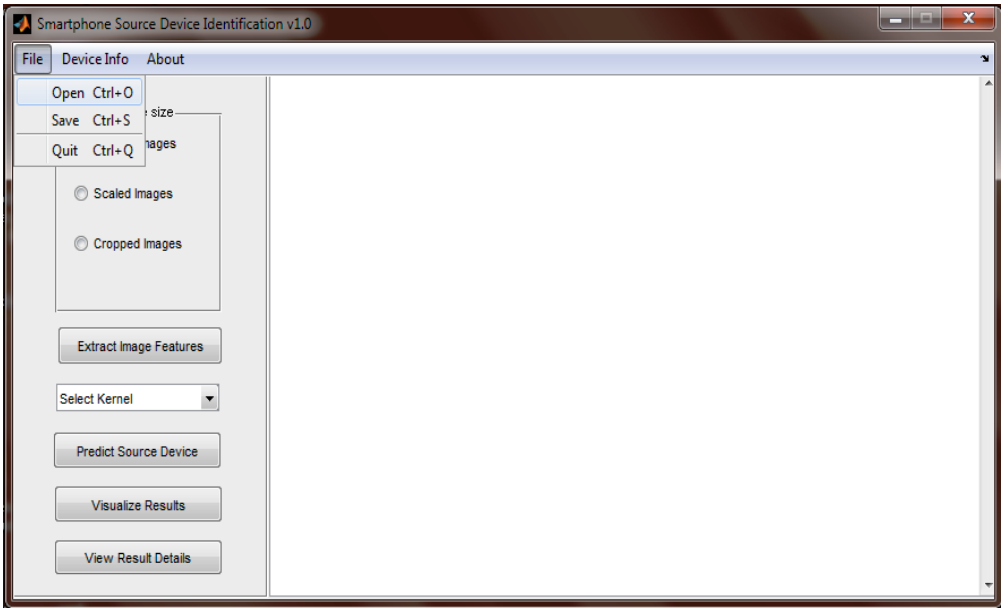


Figure 16: File Menu sub Items

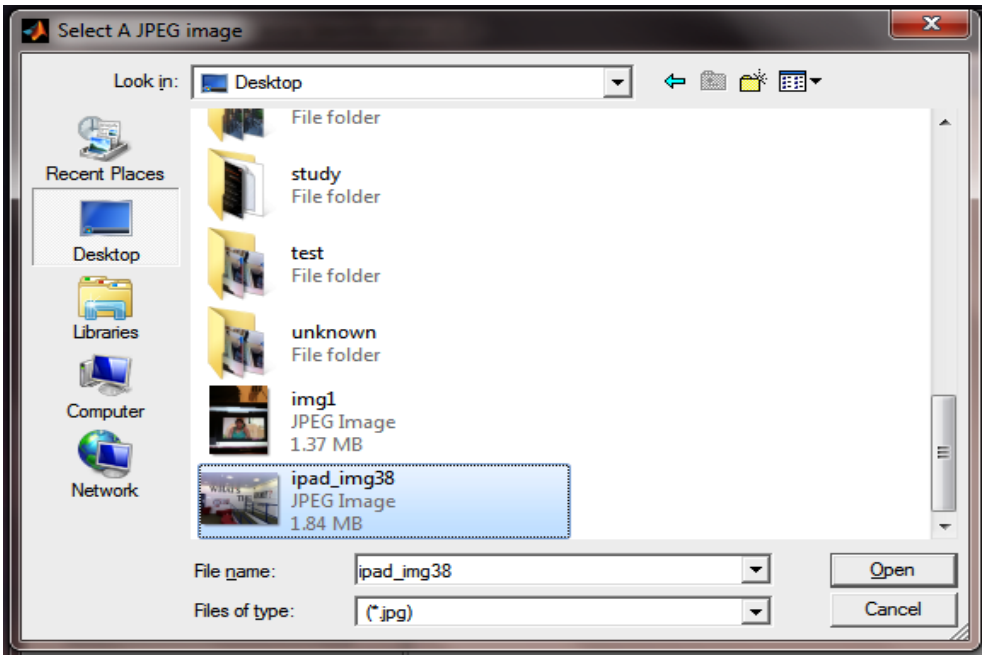


Figure 17: Browse to file window

After selecting the image, clicking on the open button displays the selected image in a different window (figure 18). The software works with only jpeg images. Not selecting the right type of file pops out an error message in figure 19. Similarly, the user is prompted should he cancel or close the select file window.

Figure 18: Image Display Window

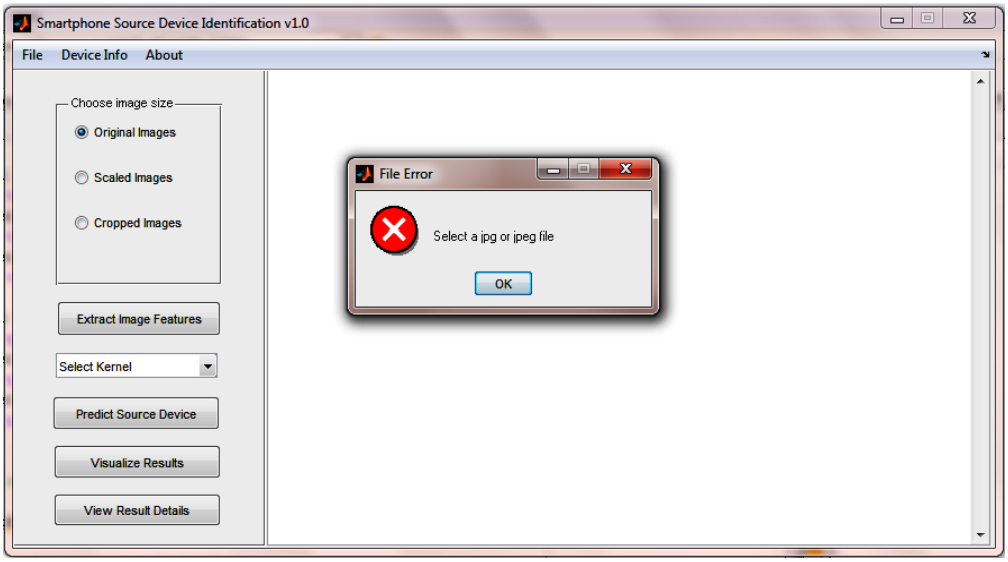


Figure 19: Wrong File type selection error message

The Device Info Menu item displays detailed information of all devices used in this project (figure 20). Each page of the device information shows 3 devices. Other devices can be viewed by clicking the small buttons at the bottom right of the window.

Figure 20: Device Information Interface

The About menu item has sub items of help (figure 21) and information about the software (figure 22).

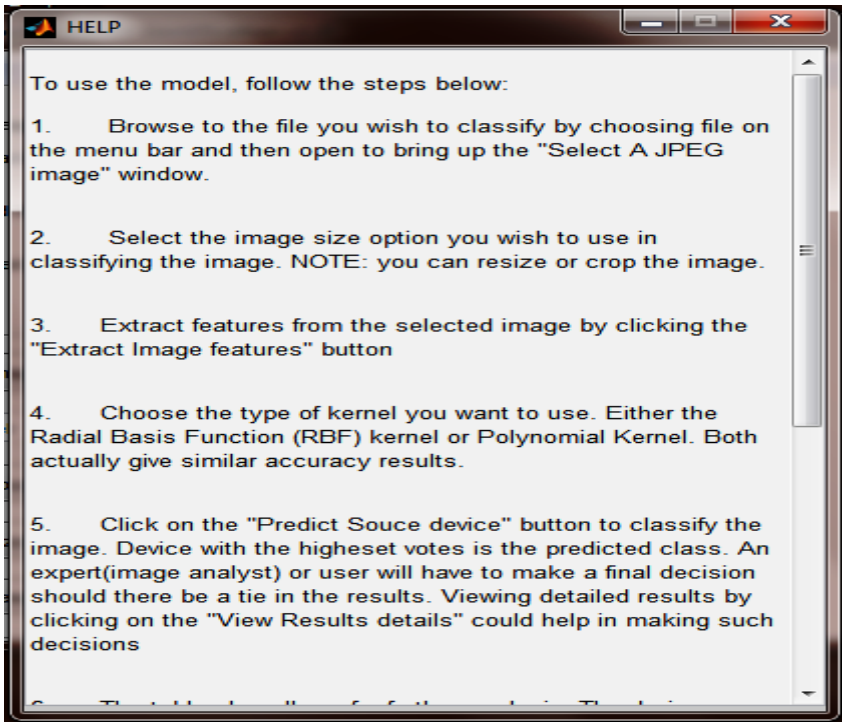


Figure 21: Help Interface

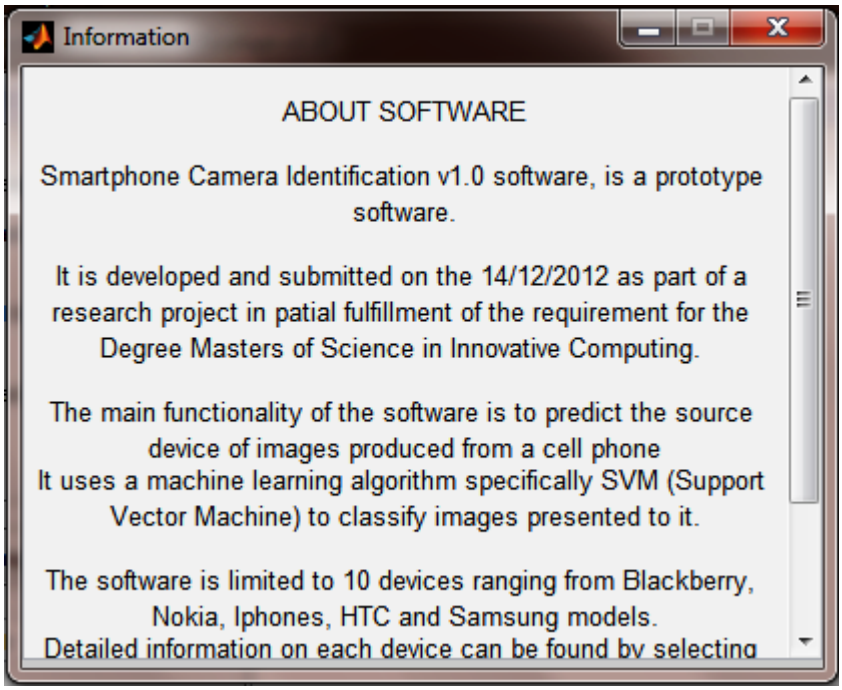


Figure 22: Software Information interface

In addition to original size images, the software is able to predict source device of images that has undergone cropping or resizing. An image size option could be selected by choosing any of the three options depending on what edit activities the image has undergone. Detailed information on cropped and resized images can be checked from the device info menu item. The user can also cross test an image (e.g. using the original size image option to test a cropped image). However, this more often does not yield accurate results because editing an image changes its features therefore an equivalent change or transformation is needed to accurately classify the edited image.

Clicking on the "Extract Image Features" button, extract features from the selected image. A small message box (figure 23) prompts the user to wait as features are being extracted. When extraction is complete, a success message box is displayed (Figure 24).

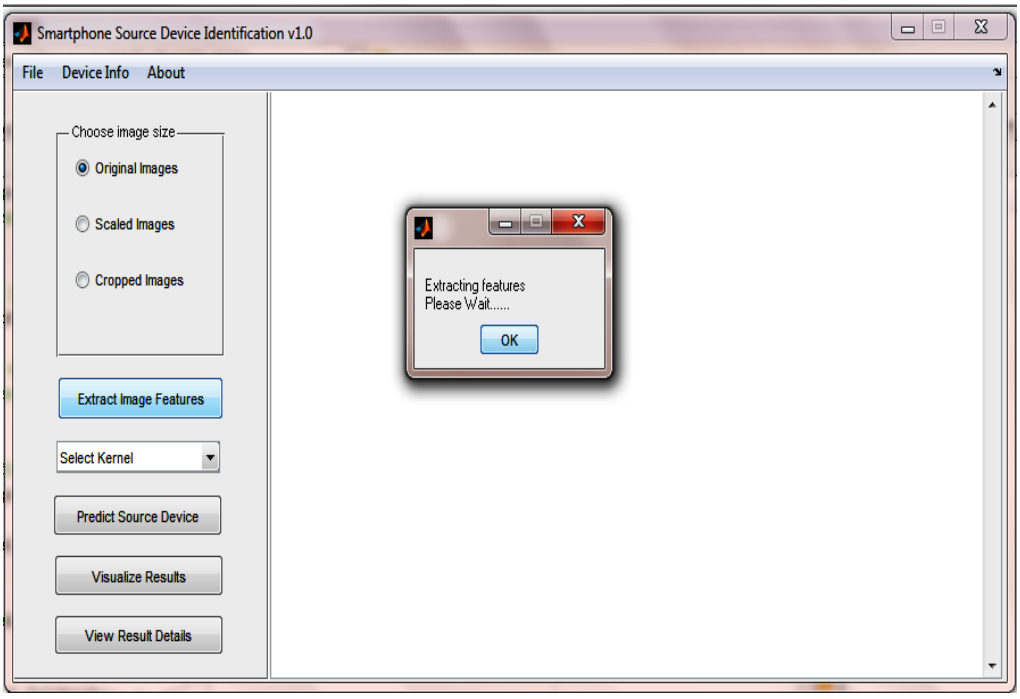


Figure 23: Extracting Features display window

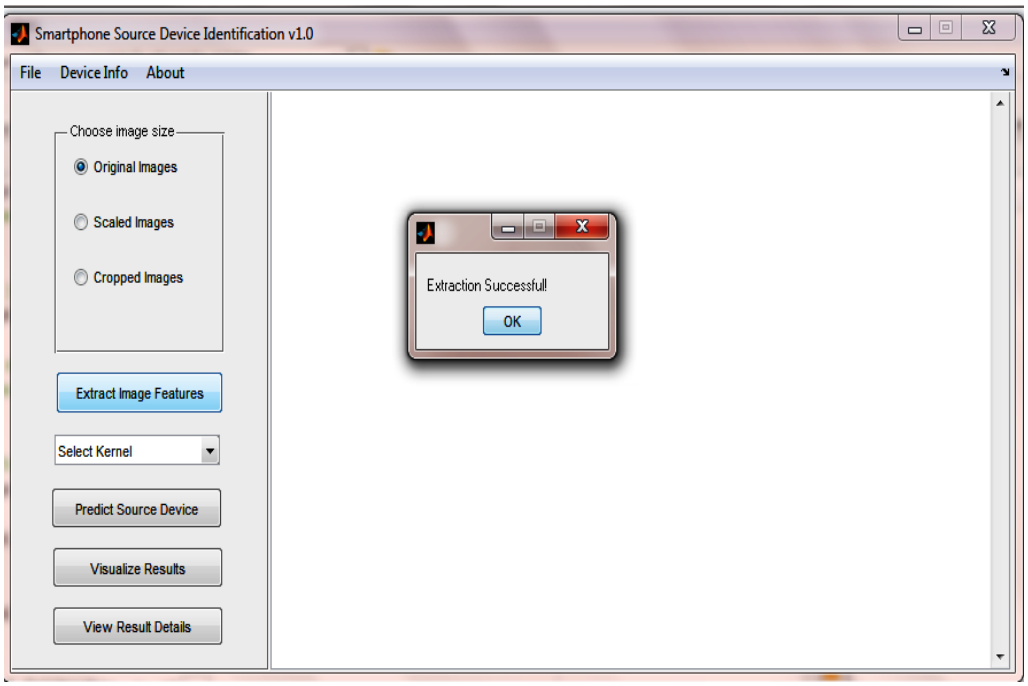


Figure 24: Feature extraction success message

A kernel must be selected for the classification task. Two options are available, the radial basis function (RBF) kernel and the polynomial kernel as shown in figure 25. Both kernels however, reports similar accurate results.

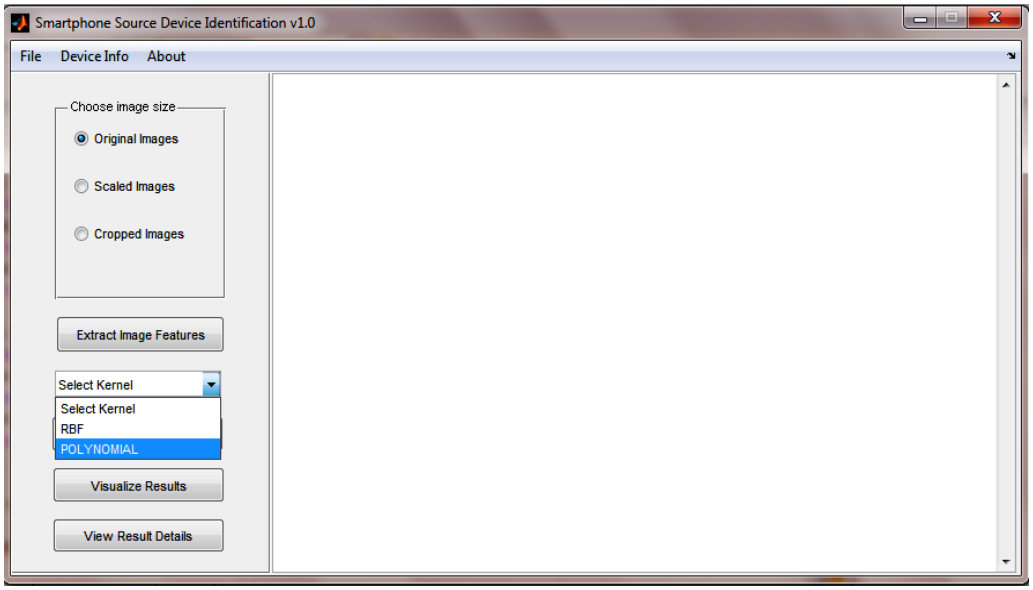


Figure 25: Selecting Kernel options

Results are displayed on the right side of the main interface as shown in figure 26 when the "Predict Source Device" button is clicked. Displayed results include the name of the selected image, selected test options for the classification and devices displayed in ascending order. Device with the highest score or votes is at the top. In this example IPAD is the most likely source device for the tested image which is true because the selected image (figure 17) is from an IPAD device.

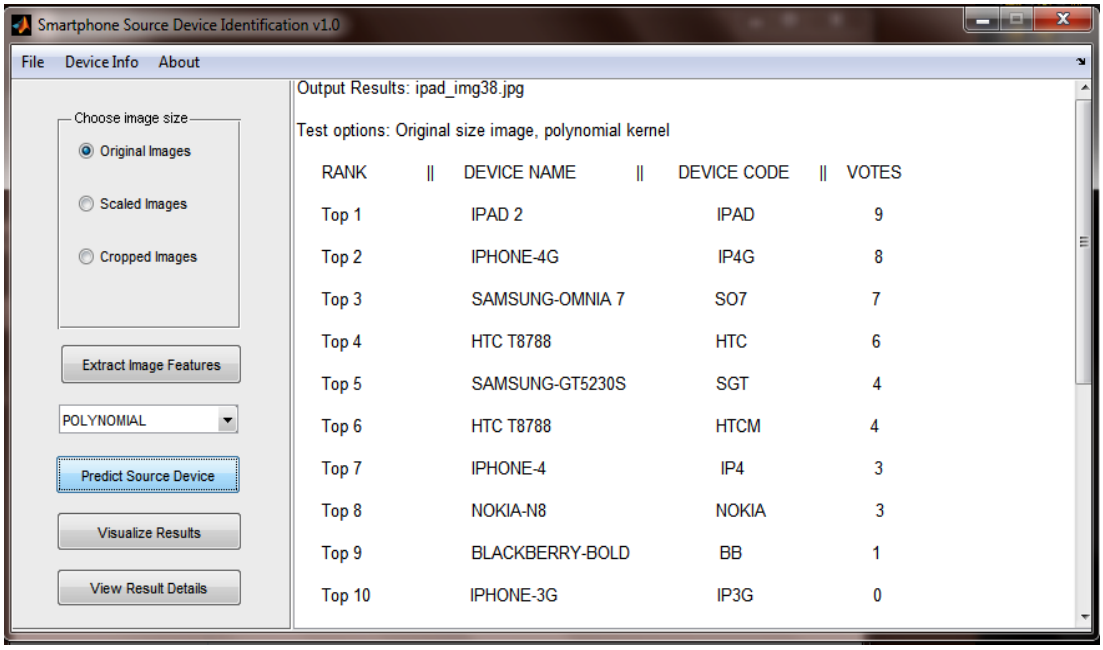


Figure 26: Prediction Result (a)

Statistics information of the results (distribution of scores/votes) is also displayed (figure 27). Information includes the average, standard deviation, estimated accuracy, standard error, margin of error, confidence coefficient of 2.821 and the confidence interval of the true accuracy. Confidence coefficient according to the t-distribution table for 98% or 97% confidence is 2.821. Alpha (α) is 0.01 and the degree of freedom (df) is 9, i.e. (10 - 1). For 93% confidence level, alpha (α) is 0.035, according to the T-table it will be 0.025 because it is closer to 0.025 than to 0.05; degree of freedom is 9, giving a coefficient of 2.262. A 92% confidence level will give a 1.833 confidence coefficient. Standard error was computed by dividing the standard deviation of the scores distribution by the square root of the sample size, in this case the number of devices (i.e. 10). The margin of error was calculated by multiplying the confidence coefficient by the standard error.

This determines the length of the confidence interval, i.e. estimated accuracy ± margin of error. In figure 27, estimated accuracy was 97% for original size images but the true accuracy for this test image lies between 97 ± 2.6346.

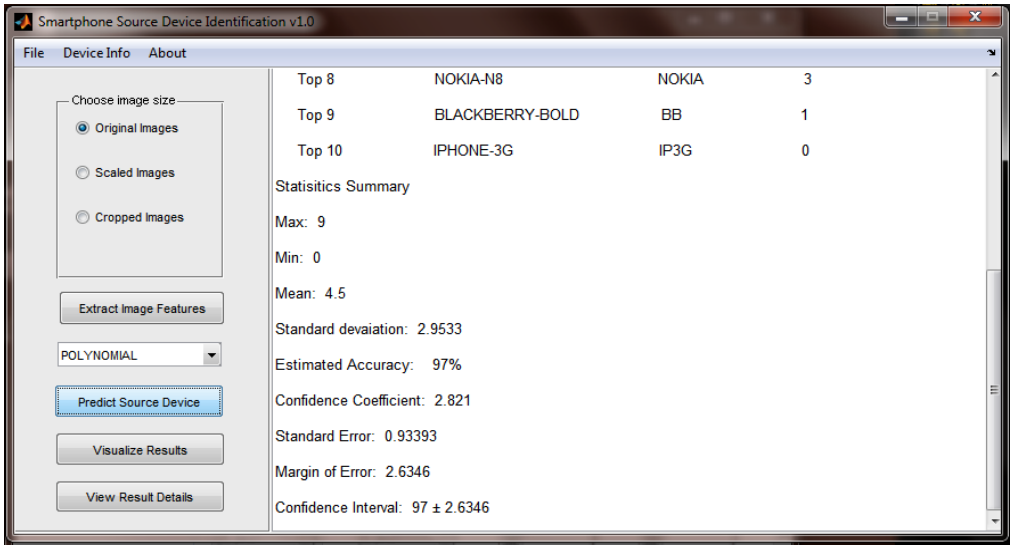


Figure 27: Prediction Results (b)

Using the radial basis function kernel, predicted class is the same as using the polynomial kernel but the difference lies in the distribution of the votes or scores (figure 28). This changes the confidence interval slightly 97 ± 2.5666. (figure 29) and comparatively is better than results of the polynomial kernel.

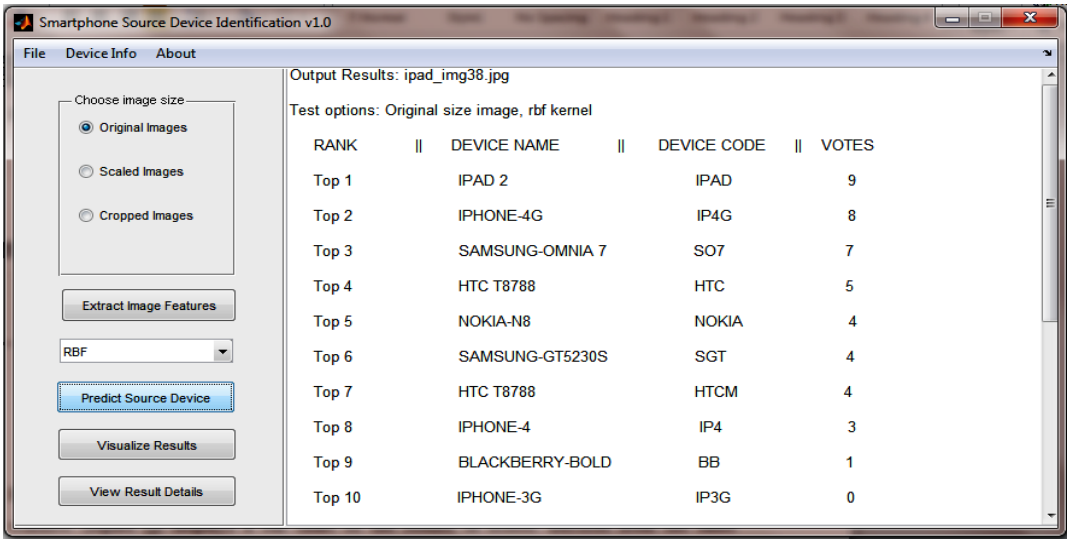


Figure 28: Prediction Results (c)

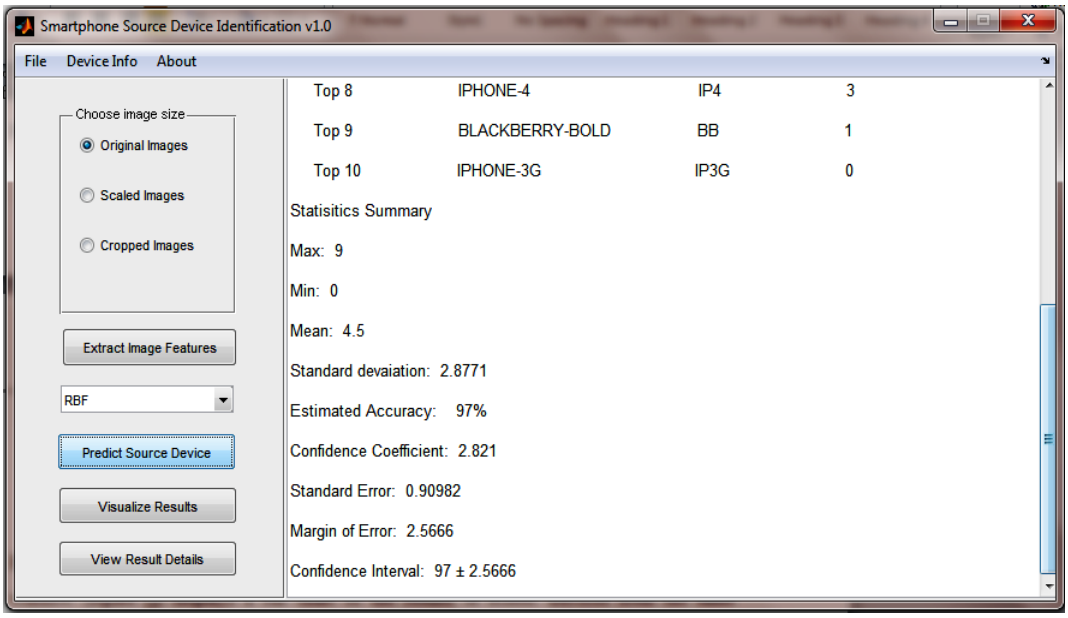


Figure 29: Prediction Results (d)

Results can also be viewed visually by clicking the "Visualize Results" button. A pop out window (figure 30) displays a bar chart of the results or scores. Devices from the same manufacturer have the same colour for example, devices from the Apple family (IPAD, IPHONE-3G, IPHONE-4, IPHONE-4G) are in red and the two HTC devices are in green, etc.

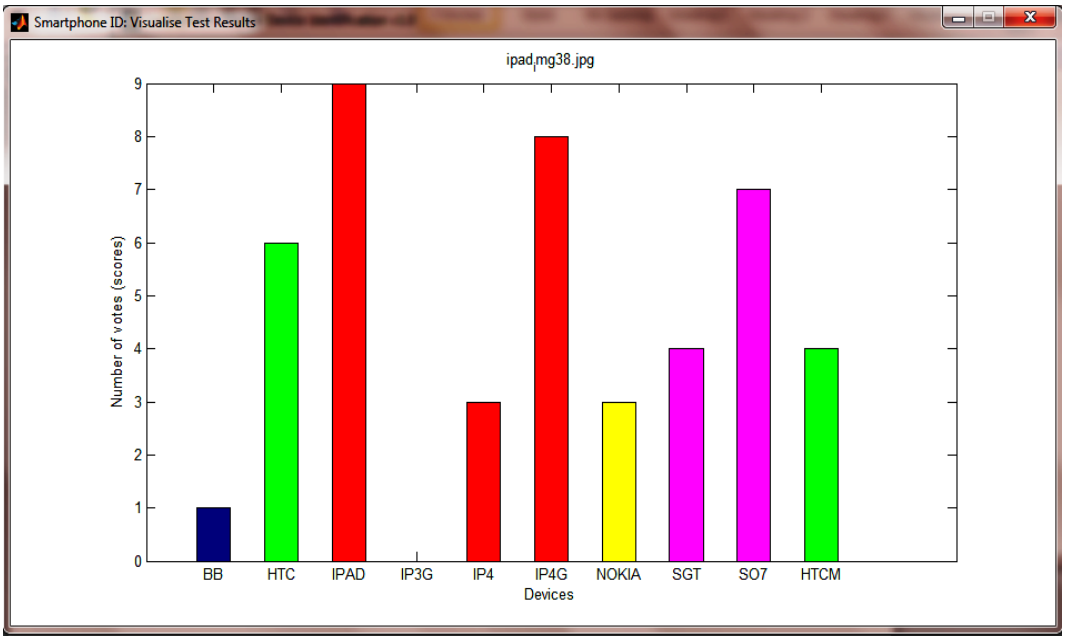


Figure 30: Visualize Result window

When a user clicks on the "View Result Details" button, a window displaying results in a table form pops out (figure 31). A vote for a class or device is represented by the "yes". An empty cell represents no vote for the device. The selected cell in figure 31 is empty. This means when the image was tested against Blackberry (BB) and HTC device, BB lost to HTC. Hence, the first cell in the HTC column had a "yes". The second cells in both BB and HTC columns are empty because they lost to the IPAD device. Therefore, the first and second cells in the IPAD column are filled with "yes". This gives the image analyst or user, detailed information of the results displayed on the main interface (figure 26). This information could be very useful especially when there is a tie in the results. A decision could be made from analysing the results when the classes (involved in the tie) were tested against each other. The total maximum votes a class could have is 9. This is indicated by the red colour where IPAD got the highest votes. No two classes can have maximum (9) votes. Ties usually occurs with 8, 7, 6 etc. votes. A green colour indicates the next highest votes (8) after the red colour. A class with 7 votes is shown in blue. Representing classes in the different colours gives the user, a quick view of the top three (could be more) devices/classes in the detailed results table. Also, an investigator may not just be interested only in the predicted class but other devices that came quite close to being predicted. This could give him other clues to work with.

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
1		yes	yes		yes	yes	yes	yes	yes	yes
2			yes			yes			yes	
3	yes	yes	yes							
4		yes	yes		yes	yes	yes	yes	yes	yes
5			yes			yes		yes	yes	yes
6		yes	yes		yes	yes				
7		yes	yes			yes		yes	yes	
8			yes			yes			yes	yes
9		yes	yes			yes	yes		yes	

Figure 31: Detailed Results

Test results can also be saved in to a .txt file by clicking the save sub menu item from the file menu or use the hot keys Ctrl + S. This opens a save dialog box for the user to type in the name of the file to be created for saving the results or overwrite an existing file. An example is shown figure 32 below.

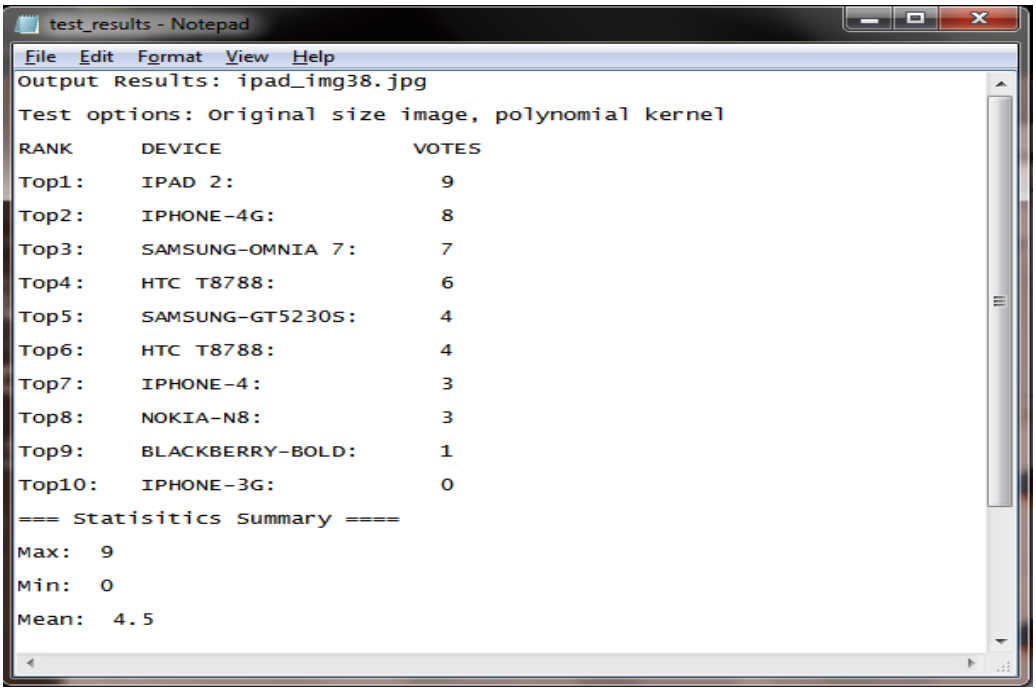


Figure 32: Test results saved in a text file

To simulate real life scenarios, few extra images (not in the database) were taken to further test the SmartPHid software. The model was able to predict the source devices correctly especially test images in their original formats but results for edited images was not pretty good.

7 CONCLUSION AND FUTURE WORK

A conclusion is drawn here with a summary of findings. The entire project is evaluated and recommendations suggested on areas that require further research.

The main objective of this project is to identify source devices of images originating from smartphones. Theoretically and practically, it is possible to identify source devices of digital images using other features other than the meta-data embedded in the image at creation. Legal issues surrounding crimes associated with smartphone devices has led to various research in the quest to discover an efficient and more accurate approach to prove the originality and authenticity of images used in the law courts as proof of evidence. Therefore the meta-data embedded in images are no longer enough to authenticate the identification of originating image devices due to the use and availability of more improved and sophisticated image processing tools for editing or deleting such information.

Various approaches that have been used to identify image source devices especially digital cameras were discussed. Some of these approaches include sensor pattern noise approach which focused on the differences in the image sensor of a device. Chromatic aberration method which uses chromatic aberration features in an image. This approach also has some relation with how different wavelengths are focused on the image sensor. Another method discussed is the lens radial distortion method. This approach aimed at distinguishing cameras based on differences in the lens of the camera. The approach used in this project considers the whole image acquisition process and therefore, tries to capture differences that may have occurred at each stage in the final image produced. Hence a set of features, broadly categorized into 3 groups (Colour features, Image Quality Features, Wavelet Features) was proposed by [11] as image features for identifying source image devices. These features were discussed into detail in the thesis.

10 smartphone devices were used to take 600 images (60 each from each camera). The 600 images were resized and cropped making a total of 1800 images for the experiments. Support Vector Machines (SVM) was the machine learning tool used for the classification process. Since SVM is basically a binary classifier, the one-vs-one approach was applied to turn it into a multi-class classifier. Applying the formula, n (n-1)/2, 45 models were built: where n is the number of classes (devices).

Experiments were conducted to analyze image features that contributed significantly in achieving high accuracy and those which did not. This information would be used in future work when the number of devices and images are increased. Only significant features will then be used which may reduce computation time. Wavelet features and Image Quality Metrics (IQM) features were the significant features reported. It was also observed that Image Quality Metrics features perform better as the training data increases. Colour features were also pretty good but in the presence of wavelet and IQM features, colour features could be dropped from the image feature set.

Experiments were also conducted on the different images sizes in the database. The three percentage spilt technique was applied in all experiments. We randomly selected 10% of total images (original size images) as the training set and the rest for testing. Then, we used 50% images in training and the rest as testing set. The third fold used 90% in training and the rest for testing. These splits were applied to experiments on cropped and resize images. It was observed that increasing the number of images used in training, increases accuracy rate. In this project the high accuracies were recorded using 90% of images in training and the rest as the test set. Original size images reported 97.5% accuracy; cropped size images reported 93% accuracy and resized images showed 92% accuracy.

In addition to the model's good performance in correctly classifying source devices, it also showed a potential of being able to classify smartphone devices of the same model from the same manufacturer. This, in future work would be improved upon (adding other significant image features) to identify a specific source device.

General performance and accuracy of the model in this project is good and acceptable comparing it to results from previous work. In depth comparison cannot be made due to differences in experimental protocols and data sets. Therefore, comparison was based mainly on reported accuracy rates and sometimes, the number of devices and total images used. With original size images, differences in results from previous work and results achieved in this project were not that significant. However, there were big differences in results on modified images. It was observed that when cropped images or resized images were tested against features from original size images, accuracy rate tends to be poor. Likewise, testing original image size features against modified image features resulted in poor accuracy. A solution was to test original size images against original image features and do same for modified images as well. This raises another important question and challenge which might be addressed in future work; How can a forensic investigator know or be sure that an image in question has

been resized, cropped or is the original image? A random guess is sure not an appropriate solution. This limitation would be addressed in future work.

Lastly, a software prototype model called SmartPHid (Smartphone Identification) to identify or predict smartphone camera devices was developed based on the outcome of the various experiments conducted. Parameters and protocols that resulted in high accuracy were used in developing the software model.

REFERENCES

[1] Yongjian Hu, Chang-Tsun Li, and Changhui Zhou, "Selecting forensic features for robust source camera identification," , 2010, pp. 506-511.

[2] Gloe et al., "An ML Perspective on Feature-Based forensic Camera Model Identification," in *DAGM 2010 Wokshop: Pattern Recognition for IT Security*, Darmstadt, Germany, 2010.

[3] M.K. Johnson and H. Farid, "Exposing Digital Forgeries through chromatic aberration," in *ACM Multimedia and Security Workshop*, Geneva, Switzerland, 2006.

[4] Lanh Tran Van, S. Emmanuel, and M. S. Kankanhalli, "Identifying Source Cell Phone using Chromatic Aberration," , 2007, pp. 883-886.

[5] Kai San Choi, Edmund Y. Lam, and Kenneth K. Y. Wong, "Source camera identification using footprints from lens aberration," in *Proceedings of the SPIE* , 2006.

[6] J. Lukas, J. Fridrich, and M. Goljan, "Digital camera identification from sensor pattern noise," *Information Forensics and Security, IEEE Transactions on*, vol. 1, no. 2, pp. 205-214, june 2006.

[7] S. Bayram, H. Sencar, N. Memon, and I. Avcibas, "Source camera identification based on CFA interpolation," , vol. 3, sept. 2005, pp. III - 69-72.

[8] Yangjing Long and Yizhen Huang, "Image Based Source Camera Identification using Demosaicking," , oct. 2006, pp. 419-424.

[9] Zhonghai Deng, A. Gijsenij, and Jingyuan Zhang, "Source camera identification using Auto-White Balance approximation," , 2011, pp. 57-64.

[10] O. Celiktutan, B. Sankur, and I. Avcibas, "Blind Identification of Source Cell-Phone Model," *#IEEE_J_IFS#*, vol. 3, no. 3, pp. 553-566, 2008.

[11] K. L. Mehdi, H. T. Sencar, and N. Memon, "Blind source camera identification," , vol. 1, 2004, pp. 709-712.

[12] H. Farid, "Detecting hidden messages using higher-order statistical models," , vol. 2, 2002, pp. II-905 - II-908 vol.2.

[13] Ismail Avcibas, İsmail Avcıbaş, Bulent Sankur, and Khalid Sayood, "Statistical Evaluation of Image Quality Measures," *Journal of Electronic Imaging*, vol. 11, pp. 206-223, 2002.

[14] Thomas Gloe, Karsten Borowka, and Antje Winkler, "Feature-Based Camera Model Identification Works in Practice," in *Information Hiding*, Stefan Katzenbeisser and Ahmad-Reza Sadeghi, Eds.: Springer Berlin / Heidelberg, 2009, vol. 5806, pp. 262-276, 10.1007/978-3-642-04431-1_19. [Online]. http://dx.doi.org/10.1007/978-3-642-04431-1_19

[15] Min-Jen Tsai and Guan-Hui Wu, "USING Image Features to Identify Camera Sources," , vol. 2, 2006.

[16] Thomas Gloe, Nicolas Cebron, and Rainer Bohme, "An ML Perspective on Feature-Based Forensic Camera Model Identification," , Darmstadt, Germany, 2010.

[17] Chih-Chung Chang and Chih-Jen Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27:1--27:27, #may# 2011. [Online]. http://doi.acm.org/10.1145/1961189.1961199

[18] Bernhard Sch\", Christopher J. C., and Alexander J. Smola, Eds., *Advances in kernel methods: support vector learning*. Cambridge, MA, USA: MIT Press, 1999.

[19] Asa B. Hur and Jason Weston, "

[20] N. Cristianini, "Support vector and kernel machines," in *18th Int. Conf. Mach. Learn*, Jun. 28, 2001.

[21] Mahesh Pal, "Multiclass Approaches for Support Vector Machine Based Land Cover Classification," *CoRR*, pp. -1--1, 2008.

[22] Rafael C. Gonzalez, Richard E. Woods, and Steven L. Eddins, *Digital Image Processing Using MATLAB*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2003.

[23] H. Du, *Data Mining Techniques and Applications: An Introduction*.: Cengage Learning, 2010. [Online]. http://books.google.co.uk/books?id=OcEZQwAACAAJ

[24] Chih-Chung Chang and Chih-Jen Lin, "

[25] MATLAB, *version 7.10.0 (R2010a)*. Natick, Massachusetts: The MathWorks Inc., 2010.

APPENDIX

TEST RESULTS ON SCALED IMAGES

Table 21: Using All Features, 90% images in training and 10% in testing

Overall Accuracy = 92%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	95.00	*	*	5.00	*	*	*	*	*	*
HTC	*	98.33	*	*	*	*	*	*	*	*
IPAD	*	*	81.67	*	*	8.33	*	*	*	*
IP3G	*	*	*	96.67	*	*	*	*	*	*
IP4	*	*	*	*	96.67	*	*	*	*	*
IP4G	*	*	8.33	*	*	83.33	5.00	*	*	*
NOKIA	*	*	*	*	*	*	98.33	*	*	*
SGT	*	*	*	*	*	*	*	96.67	*	*
SO7	*	5.00	5.00	*	*	*	*	5.00	76.67	5.00
HTCM	*	*	*	*	*	*	*	*	*	98.33

Table 22: Using only Image Quality Features, 90% images in training and 10% in testing

Overall Accuracy = 85%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	100.00	*	*	*	*	*	*	*	*	*
HTC	*	71.67	5.00	*	*	5.00	*	5.00	6.67	6.67
IPAD	*	*	86.67	*	5.00	*	*	*	*	*
IP3G	*	*	*	100.00	*	*	*	*	*	*
IP4	*	*	*	*	83.33	*	*	*	*	*
IP4G	*	*	*	*	*	80.00	*	*	*	*
NOKIA	*	*	*	*	*	*	98.33	*	*	*
SGT	*	*	5.00	*	*	5.00	*	71.67	5.00	10.00
SO7	*	5.00	8.33	*	*	8.33	5.00	6.67	65.00	*
HTCM	*	*	*	*	*	*	*	*	*	95.00

Table 23: Using only Wavelet Features, 90% images in training and 10% in testing

Overall Accuracy = 77%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	61.67	6.67	8.33	*	*	8.33	*	*	8.33	5.00
HTC	5.00	70.00	*	*	5.00	*	*	*	6.67	*
IPAD	*	5.00	45.00	*	10.00	*	10.00	8.33	8.33	6.67
IP3G	*	*	*	100.00	*	*	*	*	*	*
IP4	*	6.67	*	*	73.33	5.00	6.67	5.00	*	*
IP4G	*	*	*	*	*	93.33	*	*	*	*
NOKIA	*	*	*	*	6.67	*	86.67	*	*	*
SGT	*	6.67	*	*	*	*	*	73.33	6.67	5.00
SO7	*	*	*	*	*	*	*	*	88.33	*
HTCM	*	8.33	5.00	*	*	*	*	6.67	*	78.33

Table 24: Using only Colour features, 90% images in training and 10% in testing

Overall Accuracy = 75%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	58.33	6.67	5.00	5.00	5.00	5.00	*	*	5.00	6.67
HTC	*	98.33	*	*	*	*	*	*	*	*
IPAD	*	6.67	80.00	*	*	*	*	*	*	*
IP3G	*	*	*	85.00	*	*	*	*	*	*
IP4	5.00	6.67	*	*	61.67	*	6.67	6.67	*	6.67
IP4G	*	*	6.67	*	*	76.67	*	*	*	*
NOKIA	*	*	*	*	*	*	95.00	*	*	*
SGT	*	10.00	5.00	*	5.00	5.00	*	63.33	5.00	5.00
SO7	*	*	5.00	*	5.00	5.00	*	*	80.00	*
HTCM	5.00	10.00	5.00	6.67	5.00	*	5.00	5.00	5.00	53.33

TEST RESULTS ON CROPPED IMAGES

Table 25: Using All Features, 90% images in training and 10% in testing

Overall Accuracy = 93%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	91.7	*	5.0	*	*	*	*	*	*	*
HTC	*	98.3	*	*	*	*	*	*	*	*
IPAD	*	*	96.7	*	*	*	*	*	*	*
IP3G	5.0	*	*	91.7	*	*	*	*	*	*
IP4	*	*	*	*	96.7	*	*	*	*	*
IP4G	*	*	5.0	*	*	91.7	*	*	*	*
NOKIA	*	*	*	5.0	*	*	81.7	*	*	*
SGT	*	*	*	*	*	*	*	96.7	*	*
SO7	*	*	*	*	*	*	*	*	86.7	*
HTCM	*	*	*	*	*	*	*	*	*	100.0

Table 26: Using only Colour Features, 90% images in training and 10% in testing

Overall Accuracy = 71%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	88.33	*	5.00	*	*	*	*	*	5.00	*
HTC	6.67	63.33	5.00	*	5.00	*	5.00	5.00	*	5.00
IPAD	6.67	*	65.00	*	*	5.00	*	6.67	5.00	*
IP3G	5.00	*	5.00	73.33	*	*	6.67	*	*	*
IP4	*	*	*	*	81.67	*	*	*	*	*
IP4G	6.67	6.67	5.00	*	*	63.33	*	*	*	5.00
NOKIA	*	*	5.00	5.00	*	*	60.00	10.00	8.33	*
SGT	6.67	*	*	*	*	*	*	75.00	*	*
SO7	*	10.00	*	*	*	5.00	*	*	68.33	*
HTCM	*	5.00	10.00	*	*	5.00	*	*	*	68.33

Table 27: Using only Image Quality Features, 90% images in training and 10% in testing

Overall Accuracy = 85%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	90.00	5.00	*	*	*	*	*	*	*	5.00
HTC	*	93.33	*	*	*	*	*	*	*	*
IPAD	*	*	91.67	*	*	*	*	*	5.00	*
IP3G	*	*	*	95.00	*	*	*	*	*	*
IP4	*	*	*	*	80.00	*	*	6.67	*	*
IP4G	*	*	5.00	*	*	90.00	*	*	*	*
NOKIA	*	5.00	*	*	*	*	80.00	*	5.00	*
SGT	*	*	*	*	6.67	5.00	*	68.33	6.67	*
SO7	6.67	*	5.00	5.00	*	5.00	*	*	68.33	*
HTCM	*	*	*	*	*		*	*	*	95.00

Table 28: Using only wavelet Features, 90% images in training and 10% in testing

Overall Accuracy = 84%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	76.67	6.67	*	*	5.00	*	*	*	5.00	5.00
HTC	*	86.67	*	*	*	*	*	*	*	*
IPAD	*	*	80.00	*	*	*	8.33	*	5.00	*
IP3G	*	*	*	98.33	*	*	*	*	*	*
IP4	*	6.67	6.67	*	65.00	6.67	*	*	*	6.67
IP4G	*	*	6.67	*	*	81.67	6.67	*	*	*
NOKIA	*	*	*	*	*	*	98.33	*	*	*
SGT	*	*	*	*	*	*	*	90.00	*	*
SO7	5.00	*	*	*	*	*	6.67	*	71.67	*
HTCM	*	*	*	*	*	*	*	*	*	93.33

TEST RESULTS ON CROPPED AND RESIZE IMAGE SIZE

Table 29: Training with cropped size images and testing with resize images

Overall Accuracy = 77.7%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	84.63	4.26	*	*	*	*	*	*	*	*
HTC	*	86.85	*	*	*	*	*	*	*	7.78
IPAD	*	*	63.15	*	4.81	6.48	4.63	5.00	5.74	*
IP3G	*	*	*	87.78	*	*	*	*	*	*
IP4	*	*	*	*	72.59	*	4.63	6.48	7.22	*
IP4G	*	4.63	4.81	*	4.81	63.33	5.19	*	6.48	*
NOKIA	*	*	5.00	*	*	*	78.33	*	*	*
SGT	*	*	*	*	*	*	*	83.33	*	*
SO7	*	*	*	*	*	*	*	*	79.81	*
HTCM	*	5.74	*	*	*	*	*	*	*	77.04

Table 30: Training with Resize images and testing with cropped images

Overall Accuracy = 78.8%

	BB	HTC	IPAD	IP3G	IP4	IP4G	NOKIA	SGT	SO7	HTCM
BB	87.22	*	*	*	*	*	*	*	*	*
HTC	*	82.41	*	*	*	*	*	*	*	*
IPAD	*	*	77.78	*	*	5.19	*	*	*	*
IP3G	*	*	*	91.48	*	*	*	*	*	*
IP4	*	*	8.33	*	55.56	6.85	*	7.78	6.85	5.00
IP4G	*	*	4.81	*	*	78.15	*	*	*	*
NOKIA	*	*	*	*	*	*	81.30	*	*	*
SGT	*	*	*	*	4.07	*	5.00	73.33	4.07	*
SO7	*	*	4.07	*	*	*	*	*	77.96	*
HTCM	*	5.19	*	*	*	*	*	*	*	82.41

IMAGE FEATURES SOURCE CODE

```
function FS = features(image)

image1 = double(image);
r = image1(:,:,1);
g = image1(:,:,2);
b = image1(:,:,3);

redEnergy = sum(r(:));
greenEnergy = sum(g(:));
blueEnergy = sum(b(:));

[~, rCH rCV rCD] = dwt2(r,'haar');
[~, gCH gCV gCD] = dwt2(g,'haar');
[~, bCH bCV bCD] = dwt2(b,'haar');

red = rCH+rCV+rCD;
green = gCH+gCV+gCD;
blue = bCH+bCV+bCD;

image2 = rgb2gray(image);
noise = randn(size(image2)); noise1 = uint8(noise);
image3 = image2 + noise1;
image4 = medfilt2(image3);

FS = zeros(1, 40);

    FS(1) = mean(r(:));
    FS(2) = var(r(:),g(:));
    FS(3) = skewness(r(:));

    FS(4) = mean(g(:));
    FS(5) = var(g(:),b(:));
    FS(6) = skewness(g(:));

    FS(7) = mean(b(:));
    FS(8) = var(r(:),b(:));
    FS(9) = skewness(b(:));

    FS(10) = (greenEnergy .* greenEnergy)/(blueEnergy .* blueEnergy);
    FS(11) = (greenEnergy .* greenEnergy)/(redEnergy .* redEnergy);
    FS(12) = (blueEnergy .* blueEnergy)/(redEnergy .* redEnergy);

    FS(13) = mean(red(:));
    FS(14) = var(red(:));
    FS(15) = skewness(red(:));
    FS(16) = kurtosis(red(:));

    FS(17) = mean(green(:));
    FS(18) = var(green(:));
    FS(19) = skewness(green(:));
    FS(20) = kurtosis(green(:));

    FS(21) = mean(blue(:));
```

```
FS(22) = var(blue(:));
FS(23) = skewness(blue(:));
FS(24) = kurtosis(blue(:));

FS(25) = MaximumDifference(image2, image4);
FS(26) = MeanAbsoluteError(image2, image4);
FS(27) = MeanSquareError(image2, image4);
FS(28) = NormalizedAbsoluteError(image2, image4);
FS(29) = NormalizedMeanSquareError(image2, image4);
FS(30) = PeakSignaltoNoiseRatio(image2, image4);
FS(31) = RootMeanSquareError(image2, image4);

FS(32) = StructuralContent(image2, image4);
FS(33) = NormalizedCrossCorrelation(image2, image4);
FS(34) = CzekanowskiCorrelation(image2, image4);

FS(35) = SpectralMagnitudeDistortion(image2, image4);
FS(36) = SpectralPhaseDistortion(image2, image4);
FS(37) = SpectralPhaseMagnitudeDistortion(image2, image4);
FS(38) = BlockSpectralMagnitude(image2, image4);
FS(39) = BlockSpectralPhase(image2, image4);
FS(40) = BlockSpectralPhaseMagnitude(image2, image4);
```

FEATURE EXTRACTION SOURCE CODE (MATLAB)

```
tic;

fprintf('feature extraction...\n');

%initialize variables
BB_imgs = dir('BLACKBERRY-BOLD/*.JPG');
HTC_imgs = dir('HTC-T8788/*.JPG');
IPAD_imgs = dir('IPAD/*.JPG');
IP3G_imgs = dir('IPHONE-3G/*.JPG');
IP4_imgs = dir('IPHONE-4/*.JPG');
IP4G_imgs = dir('IPHONE-4G/*.JPG');
NOKIA_imgs = dir('NOKIA-N8/*.JPG');
SGT_imgs = dir('SAMSUNG-GT/*.JPG');
SO7_imgs = dir('SAMSUNG-OMNIA7/*.JPG');
HTCM_imgs = dir('UB-MAPS-HTC/*.JPG');


BB_features = zeros(size(BB_imgs, 1), 40);
HTC_features = zeros(size(HTC_imgs, 1), 40);
IPAD_features = zeros(size(IPAD_imgs, 1), 40);
IP3G_features = zeros(size(IP3G_imgs, 1), 40);
IP4_features = zeros(size(IP4_imgs, 1), 40);
IP4G_features = zeros(size(IP4G_imgs, 1), 40);
NOKIA_features = zeros(size(NOKIA_imgs, 1), 40);
SGT_features = zeros(size(SGT_imgs, 1), 40);
SO7_features = zeros(size(SO7_imgs, 1), 40);
HTCM_features = zeros(size(HTCM_imgs, 1), 40);

%assign labels to classes
BB_labels = ones(size(BB_imgs, 1), 1);
HTC_labels = -ones(size(HTC_imgs, 1), 1);
IPAD_labels = -ones(size(IPAD_imgs, 1), 1);
IP3G_labels = -ones(size(IP3G_imgs, 1), 1);
IP4_labels = -ones(size(IP4_imgs, 1), 1);
IP4G_labels = -ones(size(IP4G_imgs, 1), 1);
NOKIA_labels = -ones(size(NOKIA_imgs, 1), 1);
SGT_labels = -ones(size(SGT_imgs, 1), 1);
SO7_labels = -ones(size(SO7_imgs, 1), 1);
HTCM_labels = -ones(size(HTCM_imgs, 1), 1);

BB_names = cell(size(BB_imgs, 1), 1);
HTC_names = cell(size(HTC_imgs, 1), 1);
IPAD_names = cell(size(IPAD_imgs, 1), 1);
IP3G_names = cell(size(IP3G_imgs, 1), 1);
IP4_names = cell(size(IP4_imgs, 1), 1);
IP4G_names = cell(size(IP4G_imgs, 1), 1);
NOKIA_names = cell(size(NOKIA_imgs, 1), 1);
SGT_names = cell(size(SGT_imgs, 1), 1);
SO7_names = cell(size(SO7_imgs, 1), 1);
HTCM_names = cell(size(HTCM_imgs, 1), 1);
```

```
%extract features
for i = 1:size(BB_imgs, 1)
    image = imread(['BLACKBERRY-BOLD/' BB_imgs(i).name]);
    BB_features(i,:) = features(image);
    BB_names{i} = ['BLACKBERRY-BOLD/' BB_imgs(i).name];
end

for i = 1:size(HTC_imgs, 1)
    image = imread(['HTC-T8788/' HTC_imgs(i).name]);
    HTC_features(i,:) = features(image);
    HTC_names{i} = ['HTC-T8788/' HTC_imgs(i).name];
end

for i = 1:size(IPAD_imgs, 1)
    image = imread(['IPAD/' IPAD_imgs(i).name]);
    IPAD_features(i,:) = features(image);
    IPAD_names{i} = ['IPAD/' IPAD_imgs(i).name];
end

for i = 1:size(IP3G_imgs, 1)
    image = imread(['IPHONE-3G/' IP3G_imgs(i).name]);
    IP3G_features(i,:) = features(image);
    IP3G_names{i} = ['IPHONE-3G/' IP3G_imgs(i).name];
end

for i = 1:size(IP4_imgs, 1)
    image = imread(['IPHONE-4/' IP4_imgs(i).name]);
    IP4_features(i,:) = features(image);
    IP4_names{i} = ['IPHONE-4/' IP4_imgs(i).name];
end

for i = 1:size(IP4G_imgs, 1)
    image = imread(['IPHONE-4G/' IP4G_imgs(i).name]);
    IP4G_features(i,:) = features(image);
    IP4G_names{i} = ['IPHONE-4G/' IP4G_imgs(i).name];
end

for i = 1:size(NOKIA_imgs, 1)
    image = imread(['NOKIA-N8/' NOKIA_imgs(i).name]);
    NOKIA_features(i,:) = features(image);
    NOKIA_names{i} = ['NOKIA-N8/' NOKIA_imgs(i).name];
end

for i = 1:size(SGT_imgs, 1)
    image = imread(['SAMSUNG-GT/' SGT_imgs(i).name]);
    SGT_features(i,:) = features(image);
    SGT_names{i} = ['SAMSUNG-GT/' SGT_imgs(i).name];
end

for i = 1:size(SO7_imgs, 1)
    image = imread(['SAMSUNG-OMNIA7/' SO7_imgs(i).name]);
    SO7_features(i,:) = features(image);
    SO7_names{i} = ['SAMSUNG-OMNIA7/' SO7_imgs(i).name];
end
```

```
for i = 1:size(HTCM_imgs, 1)
    image = imread(['UB-MAPS-HTC/' HTCM_imgs(i).name]);
    HTCM_features(i,:) = features(image);
    HTCM_names{i} = ['UB-MAPS-HTC/' HTCM_imgs(i).name];
end

%Normalize the features sets
mu = mean([BB_features; HTC_features; IPAD_features; IP3G_features;
IP4_features; IP4G_features; NOKIA_features; SGT_features; SO7_features;
HTCM_features]);
sigma = std([BB_features; HTC_features; IPAD_features; IP3G_features;
IP4_features; IP4G_features; NOKIA_features; SGT_features; SO7_features;
HTCM_features]);

normalized_BB_features = zeros(size(BB_features));
normalized_HTC_features = zeros(size(HTC_features));
normalized_IPAD_features = zeros(size(IPAD_features));
normalized_IP3G_features = zeros(size(IP3G_features));
normalized_IP4_features = zeros(size(IP4_features));
normalized_IP4G_features = zeros(size(IP4G_features));
normalized_NOKIA_features = zeros(size(NOKIA_features));
normalized_SGT_features = zeros(size(SGT_features));
normalized_SO7_features = zeros(size(SO7_features));
normalized_HTCM_features = zeros(size(HTCM_features));

for i = 1:size(BB_features, 1)
    normalized_BB_features(i,:) = (BB_features(i,:)-mu)./sigma;
end

for i = 1:size(HTC_features, 1)
    normalized_HTC_features(i,:) = (HTC_features(i,:)-mu)./sigma;
end

for i = 1:size(IPAD_features, 1)
    normalized_IPAD_features(i,:) = (IPAD_features(i,:)-mu)./sigma;
end

for i = 1:size(IP3G_features, 1)
    normalized_IP3G_features(i,:) = (IP3G_features(i,:)-mu)./sigma;
end

for i = 1:size(IP4_features, 1)
    normalized_IP4_features(i,:) = (IP4_features(i,:)-mu)./sigma;
end

for i = 1:size(IP4G_features, 1)
    normalized_IP4G_features(i,:) = (IP4G_features(i,:)-mu)./sigma;
end

for i = 1:size(NOKIA_features, 1)
    normalized_NOKIA_features(i,:) = (NOKIA_features(i,:)-mu)./sigma;
end

for i = 1:size(SGT_features, 1)
    normalized_SGT_features(i,:) = (SGT_features(i,:)-mu)./sigma;
end
```

```
for i = 1:size(SO7_features, 1)
    normalized_SO7_features(i,:) = (SO7_features(i,:)-mu)./sigma;
end

for i = 1:size(HTCM_features, 1)
    normalized_HTCM_features(i,:) = (HTCM_features(i,:)-mu)./sigma;
end

 %save extracted features

 save('afeatures1.mat', 'normalized_BB_features',
'BB_labels','BB_names','normalized_HTC_features', 'HTC_labels',
'HTC_names','normalized_IPAD_features', 'IPAD_labels',
'IPAD_names','normalized_IP3G_features', 'IP3G_labels',
'IP3G_names','normalized_IP4_features', 'IP4_labels',
'IP4_names','normalized_IP4G_features', 'IP4G_labels',
'IP4G_names','normalized_NOKIA_features',
'NOKIA_labels','NOKIA_names','normalized_SGT_features', 'SGT_labels',
'SGT_names','normalized_SO7_features', 'SO7_labels',
'SO7_names','normalized_HTCM_features', 'HTCM_labels', 'HTCM_names');

save('norm','mu','sigma')


toc;

tic;
```