Master of Science in Computer Science
February 2017

Splicing Forgery Detection and the Impact of Image Resolution

Vishnu Manasa Devagiri

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona Sweden

Contact Information:
Author(s):
Vishnu Manasa Devagiri
vida15@student.bth.se

University advisor:
Dr. Abbas Cheddad
Dept. Computer Science and Engineering

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

ABSTRACT

Context. There has been a rise in the usage of digital images these days. Digital images are being used in many areas like in medicine, wars, etc. As the images are being used to make many important decisions, it is necessary to know if the images used are clean or forged. In this thesis we have considered the area of splicing forgery. In this thesis we are also considering and analyzing the impact of low-resolution images on the considered algorithms.

Objectives. Through this thesis we try to improve the detection rate of splicing forgery detection. We also examine how the examined splicing forgery detection algorithm works on low-resolution images and considered classification algorithms (classifiers).

Methods. The research methods used in this research are Implementation and Experimentation. Implementation was used to answer the first research question i.e., to improve the detection rate in splicing forgery. Experimentation was used to answer the second research question. The results of the experiment were analyzed using statistical analysis to find out how the examined algorithm works on different image resolutions and on the considered classifiers.

Results. One tailed Wilcoxon signed rank test was conducted to compare which algorithm performs better, the T+ value obtained was less than T_o so the null hypothesis was rejected and the alternative hypothesis which states that Algorithm 2 (our enhanced version of the algorithm) performs better than Algorithm 1 (original algorithm), is accepted. Experiments were conducted and the accuracy of the algorithms in different cases were noted, ROC curves were plotted to obtain the AUC parameter. The accuracy, AUC parameters were used to determine the performance of the algorithms.

Conclusions. After the results were analyzed using statistical analysis, we came to the conclusion that Algorithm 2 performs better than Algorithm 1 in detecting the forged images. It was also observed that Algorithm 1 improves its performance on low resolution images when trained on original images and tested on images of different resolutions but, in the case of Algorithm 2, it performance is improved when trained and tested on images of same resolution. There was not much variance in the performance of both of the algorithms on images of different resolution. Coming to the classifiers, Algorithm 1 improves its performance on linear SVM whereas Algorithm 2 improves its performance when using the simple tree classifier.

Keywords: Splicing Forgery, Machine Learning, Image Processing.

ACKNOWLEDGEMENTS

Initially, I would like to acknowledge and thank Dr. Abbas Cheddad, my thesis supervisor. His continuous support and guidance helped me in conducting this thesis successfully.

Secondly I would like to thank “Image Analysis Processing & Protection Group” for providing the partial source code of the algorithm considered to be improved in this thesis. I would also like to acknowledge A. De Rosa, M. Fontani, M. Massai, A. Piva, and M. Barni the authors of the paper which proposed this algorithm.

I would also like to thank “Columbia Uncompressed Image Splicing Detection Evaluation Dataset” group for providing the dataset which is used in this thesis for training and testing the models.

Lastly, I would like to thank my family and friends for their love and support in all phases of my life.

LIST OF FIGURES

List of Tables

LIST OF ABBREVIATIONS

SVM	Support Vector Machine
ANN	Artificial Neural Networks
DCT	Discrete Cosine Transformation
DWT	Discrete Wavelet Transformation
IQM	Image Quality Metrics
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives
ROC	Receiver Operating Characteristics
AUC	Area Under the ROC Curve
SD	Standard Deviation
TPR	True Positive Rate
FPR	False Positive Rate

CONTENTS

1 INTRODUCTION AND BACKGROUND

The number of digital images in use has increased considerably these days, be it due to social media, or electronic media like digital newspapers, etc. Along with the rise in the usage of digital images there has also been many queries about the integrity of these images as we are having easy access to many image manipulation tools like Photoshop these days. It is necessary to know if an image is clean or tampered with as the images are sometimes being used to make important decisions [1]. The images are also used to examine the state of war between two countries as stated by Abhishek et al [2]. Active image tampering techniques are those which can be detected by using methods such as watermarking or signatures [3]. Watermarking necessitates that the cameras used to take photographs should be equipped with certain features such that the image contains a watermark or signature, which is later used to authenticate the image. This information is not available on all the images as not all cameras are equipped with this technology [1]. These methods where we do not have any of this information like signature or watermarks are known as passive techniques [4]. These techniques are also known as blind techniques. As these passive methods do not require any prior information, many researches deal with these areas [3].

Image forgery or tampering can also be classified as either intra or inter image. Inter image tampering is an image manipulation technique in which only a single image is used where as in Intra image tampering two or more images are used to manipulate an image. As the area of Image forgery is vast we have considered a smaller area of splicing forgery to conduct our research. We have considered splicing forgery as this is one of the mostly commonly used forgery technique to manipulate images [2]. Splicing forgery detection can also be classified as a passive technique since in this type of forgery detection, features such as digital watermarking, signatures are not considered. As this technique do not require prior information about the images these can be used on all types of images. Splicing forgery is a type of inter image tampering technique. In splicing forgery two or more images are combined together to form a new image [1][4]. In our work we try to improve an existing algorithm in detecting whether a given image is clean or forged. The impact of the image resolutions using the studied algorithms on classifiers has also been analyzed.

The research methods used in this thesis are implementation and experimentation. Implementation is used to answer RQ1 i.e., to improve the detection rate of the existing algorithm. Experimentation was used to answer RQ2 i.e., to check the impact of the image resolution and classifiers on the considered algorithm.

In this research we have used the fields of image processing and machine learning to classify an image into either a clean or a forged one. Image processing is used to extract features from an image which could be helpful while training a model using the machine learning algorithms. The machine learning is used to generate a model which learns about the different features of the images and would be able to predict when a new image has been subjected to forgery. Hence it could detect whether the new image is either clean or not based on the model built in the training phase. Authors of Ref.[5][6][7] have also classified their images using machine learning algorithms.

1.1 Background

This sub-section gives a brief description of areas of study used in this research. In this section description of image forgery, feature extraction and machine learning is given. In the

Image forgery sub-section, a brief description of the types of image forgery and their examples is highlighted.

1.1.1 Image Forgery

When an image has been manipulated or changed to achieve the desired results then the image is said to have been forged or tampered. This can generally be done in two ways, one by manipulating the image without using any other new images and the other by combining two or more images to acquire the desired result. The first one can be referred to as an intra image tampering and the second one as inter image tampering. These two forgery techniques can be further divided into many other types, some of them are described below.

Intra Image tampering: Some of the most common techniques used in this category are copy-move, contrast-enhancement, image retouching, etc. In the copy move forgery a part of the image is cut and pasted in the same image at a favored location so as to obtain the desired effect in the picture [8]. In the contrast- enhancement the contrast of the image is changed to obtain the desired results. Sometimes contrast-enhancement is just used to hide the tampering done on the image. In image retouching the features of the image are modified so as to obtain the desired results [4]. The below figures - 1.1 and 1.2, have been obtained from Ref.[9]. In Figure 1.2 the person is hidden by pasting the region of the background on the person.

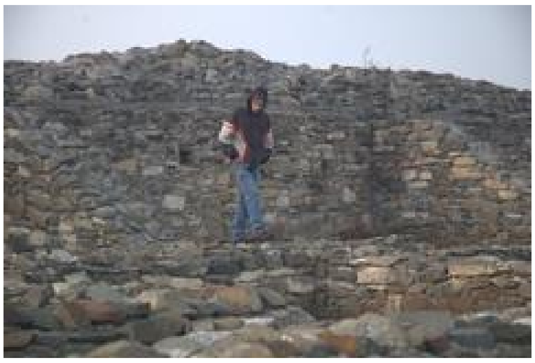


Figure 1.1: Original Image.

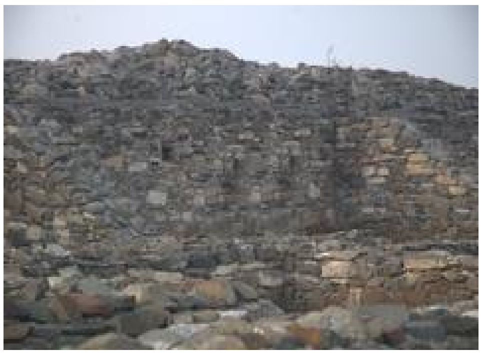


Figure 1.2: Image after copy move forgery.

Inter Image tampering: The most common method used in this type of forgeries is the splicing forgery. In splicing forgery, an image is manipulated by pasting portions of different images onto an image to obtain the desired output on the image. Figures 1.3 and 1.4 show an example of splicing forgery, they are retrieved from the dataset [10] used in this research.

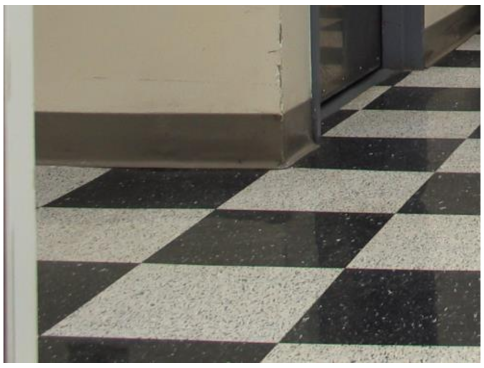


Figure 1.3: Original Image.

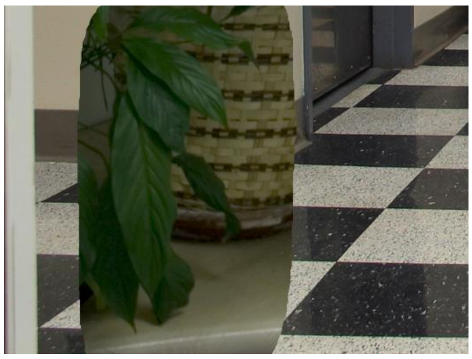


Figure 1.4: Image after splicing forgery.

1.1.2 Feature Extraction

As stated earlier, image processing was used to extract the features from the images. We have used Ref.[11] as a reference guide to learn more about image processing using Matlab® and the types of features that could be extracted from the images. Endless of features can be extracted from an image and it is extremely important to extract the right set of features to obtain good results. The features extracted from the images could be either first-order statistics or second-order statistics. First order statistics are the most commonly used. First- order statistics are obtained by analyzing the histogram i.e., calculating the average, variance, etc. of a histogram [5]. In this thesis, the second-order statistics are obtained from the co-occurrence matrix. Unlike the second-order statistics it is generally easy to manipulate the first-order statistics while applying counter forensic techniques so as to hide the tampering [5].

In this research features were extracted from an RGB color image after converting it into YCbCr color space and then applying discrete wavelet decomposition. A more detailed description of the process used in this research is described in chapter 4 (Approach section).

1.1.3 Machine learning

As stated, many authors were using machine learning algorithms to classify the images. In this research we have also taken the help of these algorithms for classification. After completing the feature extraction process, the data extracted is trained on different machine learning algorithms available in the Matlab® Tool-box and we have considered linear SVM and Simple Tree models for classification of the test dataset as their predicted accuracy was the best.

1.1.3.1 Classifier

A classifier is an algorithm which is used to train a model by using the training data. A classifier generates a model after it has been successfully trained by the training dataset. This model is then later used to classify the test data into their respective classes. The performance of the classifier is measured using various parameters like the accuracy, training time, F-measure, sensitivity, specificity, precision. We used Ref.[12] to learn more about these parameters.

Confusion matrix: A confusion matrix is generally used to describe how well a classifier has performed. It clearly shows the number of instances correctly classified, the number of instances which have been wrongly classified. It clearly states the actual class of the instance that it belongs to and the class to which it has been classified into. If Class 0 is a positive class and Class 1 is a negative class, then the classifier's confusion matrix could be represented as in table 1.1. In the table, TP stands for true positives i.e., the number of positive instances correctly classified; FP stands for false positives i.e., the number of negative instances classified as positive by the classifier; FN stands for false negatives i.e., the number of positive instances classified as negative; TN stands for true negatives i.e., the number of negative instances classified as negative.

	Predicted Class 0	**Predicted Class 1**
Actual Class 0	TP	FN
Actual Class 1	FP	TN

Table 1.1: Confusion Matrix Structure.

Accuracy: This is the most common measure used to state how well a classifier has performed. It describes how well the classifier has classified the instances. Accuracy is the ratio of the number of correctly classified instances to that of the total number of instances.

Equation 1:

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ Number\ of\ Instances} = \frac{Number\ of\ Instances\ correctly\ classified}{Total\ Number\ of\ Instances}$$

Training Time: It is the time taken by the classifier to generate a model, which is later used to classify the instances whose class label is not known.

Sensitivity: Sensitivity is also known as the true positive rate or recall. It states the rate at which the positive instances are correctly classified. The formulae to calculate the sensitivity is stated in Equation 2.

Equation 2: $Sensitivity = \frac{TP}{TP+FN}$

Specificity: It is also known as the true negative rate. This parameter states the rate at which the negative class label which has been classified correctly. The formulae to calculate the sensitivity is stated using Equation 3.

Equation 3: $Specificity = \frac{TN}{TN+FP}$

Precision: Precision is the ratio of correctly classified positive instances to that of the total number of instances that have been classified as positive by the classifier. The formulae to calculate precision is stated in Equation 4.

Equation 4: $Precision = \frac{TP}{TP+FP}$

F-measure: This is also known as F1 score or F score. F-measure is the harmonic mean of precision and sensitivity.

Equation 5: $F - Measure = 2 \times \frac{(Precision \times Sensitivity)}{(Precision + Sensitivity)}$

1.2 Structure of the Thesis

The other sections present in this document are related work, aims and objectives, approach, methodology, results, analysis and discussion, followed by conclusion and future work. The related work section describes about the previous work done in this field. The aims and objectives section states the aims, objectives and the research questions which were answered through this thesis. In the approach section the datasets, tools used were described along with the algorithms considered in this research. Next comes the methodology section through which the methods used to conduct research were discussed in detail. In the results section, the results obtained through the implementation and experimentation are documented and explained using graphs and tables. This is followed by the analysis and discussion section where the results obtained are analyzed and the research questions are answered. At the end of the document we have the conclusion and future work section which states the conclusions of this research followed by the potential areas in which this research could be improved.

2 RELATED WORK

This section deals with the previous work conducted in this area of research i.e. splicing forgery in digital images. The researchers have researched about different techniques so as to find a better way to detect image forgery. The text below describes some of the contributions of researchers in this field.

The authors of Ref.[5] [13] have considered the aspect of contrast enhancement in images to deal with forgery detection. In Ref.[5] the authors have proposed a method which can detect image tampering even if it is hidden using the counter-forensic technique of contract enhancement, whereas in Ref.[13], contrast enhancement has been used to detect the image forgery.

The authors of Ref.[5] have considered second order statistics of the co-occurrence matrix to detect image tampering even when the examined image has undergone some contrast enhancement to manipulate the histogram. Experiments are used to test the proposed algorithm[5]. Generally speaking, histograms are widely used to detect if an image has been tampered with or not. In natural images, the histogram is usually smooth without any abrupt peaks in it if the image is not tampered with. Contrast enhancement is one of the commonly used counter forensics method. In this method, the contrasts of the image are altered in such a way that the histogram would look normal. The authors have also stated that there are many papers which have proposed several anti-forensics methods to manipulate the histogram based detection techniques. To overcome such anti forensic approaches, this paper has proposed an algorithm using co-occurrence matrix which can work effectively even if the contrast enhancement technique has been used to manipulate the forged image. The authors of the paper have conducted three experiments to check for the efficiency of their algorithm. In the first experiment, they implemented the algorithm on the image which is manipulated using Gama corrections and tested if it could detect the tampering. Thereafter they tested the algorithm on the images where counter forensic methods have been used to hide the tampering. In the last experiment conducted the authors checked how well the algorithm can be used to sort the images into clean and tampered ones using the Support Vector Machine (SVM) algorithm.

The authors of Ref.[13] have stated that generally the image tampering which is done by contrast enhancement can be detected based on the sudden peaks that occur in a histogram due to image manipulation. But the histograms could be easily manipulated using counter forensic techniques so in this paper the authors have tried to focus on other features and considered the similarities in the channels of high frequency components to detect a forgery in an image. It is stated that there will be changes in the channels due to contrast enhancement which leads to the differences in the channels hence notifying that contrast enhancement has been used to hide the tampering. The drawback of the method is that the forgery could still be unidentified if all the three channels i.e. R, G, B have been enhanced equally, but they also state that these kind of situations does not occur frequently when we are considering splicing image forgery (cut and paste) which the authors have considered. Although, their algorithm had been tested and validated under both forensic and counter forensic conditions, one of the drawbacks is that it does not work well with the images which have undergone JPEG compression.

The authors of Ref. [14] have designed an algorithm which can detect the forged regions of a tampered image taken through digital cameras. They have considered that the color filter array (CFA) values are obtained while taking the image and hence made use of the demosaicking algorithm to detect the forged regions. In demosaicking algorithm, the values in one part of the image can be derived from its adjacent values. The algorithm takes into consideration each 2*2 block of the image and applies a method to check if that region of the image has been tampered with or not. Their algorithm is mainly based on the fact that the demosaicking values of an authentic image are regular, but sometimes these assumptions may lead to false alarms. The used CFA contains strips of red and green filters in one row followed by the blue and green ones in the second. Some of the drawbacks of this algorithm is that it cannot accurately detect forgeries in images with higher compression levels and the algorithm is not apt for detecting the copy-move forgeries, it also does not work well in flat or sharp edges.

The authors of Ref.[15][16] have used geometry based methods and considered the relative sizes of the objects in the image with respect to each other so as to detect the forged ones from the clean ones. The authors of the two papers have also claimed that their algorithm produces good results even for low resolution images.

In Ref.[15] the ratio of the heights of two objects in the same image is considered to detect the forgery, the authors have proposed a perspective constraint method through which this ratio is obtained using the vanishing line (a geometry term). This helps to calculate the ratio without using the parameters of camera. The image is classified as a forged one and the manipulated region is detected if the ratio had exceeded a predefined threshold value.

On the other hand, the authors of the Ref.[16] have proposed a similar model to that of the Ref.[15] but have stated that it works better for images with tilt and roll. In their proposed method, the user has to do some of the important things manually, like selecting the objects and the vanishing lines; which may sometimes not be as efficient as an automatic procedure. This is because there can be mistakes made by humans and the second reason is that he/she should have some knowledge about the tool and how it works in order to perform the tasks (i.e., training). They have also stated that they are working on this to avoid humans make such decisions.

The authors of Ref.[6] have presented an algorithm which extracts features using; *Image Quality Metrics* (IQMs) from the images and then used machine learning techniques to create a model which can be used to classify an image as either forged or a clean one. They have used Artificial Neural Network (ANN) to classify the images. The authors have extracted ten quality metrics from the images which are later used in the creation of the model used for the classification of the images. The authors have also used a backpropagation algorithm to reduce the error rate. The error rate is calculated and used by the backpropagation algorithm to modify the network weights as required. The ANN algorithm is implemented using the ANN Toolbox in Matlab®. The authors have stated that their proposed algorithm's performance is affected by how well the model is trained; which is a common machine learning issue. The authors have also stated that the algorithm needs to be further improved in order to obtain more accurate classification of images [6].

The authors of Ref.[2] have stated how image forgeries can be detected, they have stated that the forgery detection techniques can be classified into two classes named active and passive methods. The authors have considered the passive methods over the active ones in their research. They claimed that their algorithm performs better than the previous ones as they have minimized the number of computations required for the implementation of the algorithm. The authors have used multi-level wavelet decomposition and block matching which is followed by creating a map of duplicate regions in their algorithm to classify images as either forged or clean. They have also stated that their algorithm is 87.75% accurate. The major setback of the algorithm which the authors have also mentioned, is that the algorithm cannot perform well when counter- forensics methods are applied to the image after tampering [2].

The authors of Ref.[7] have also utilized a machine learning algorithm to classify the images under scrutiny into two categories; forged or clean. Like in Ref.[5] they have also used SVM to classify the images. The features extracted in this algorithm include both shape and color based attributes. On these values they constructed a model by using the SVM classifier which is later used to distinguish between forged and clean images. The authors have stated that their algorithms use both color and the boundaries attributes to detect forgeries which is better when compared to the algorithms which use a single type of these attributes. The proposed algorithm is claimed to be 74% accurate and is automated to a maximum extent [7].

Through Ref.[4], the authors discuss the different types of forgeries like copy-move, splicing, retouching, etc. The authors have first described each of the existing forgery algorithms along with available methods to detect them. In the copy-move forgery, a part of the image is cut and pasted in the same image so as to manipulate the image in the desired way. These forgeries are generally detected using the brute force method or the block based matching where the image is inspected by dividing it into blocks and then examining each block. It is also stated that the block based methods perform better than the ones using the brute force attack. They have also described about various types of methods which come under these categories. They have classified the methods used to splicing forgery as boundary based and region based. In boundary based methods, the forgery is detected based on the presence of irregular edges. Whereas in region based methods, it is detected by comparing the consistency of the image with respect to its statistical generated model. The authors have also stated that many algorithms are not effective for geometrical transformations, and have problems like high complexity, less reliability and less accuracy.

The authors of Ref. [17][18] have used the noise present in the image to detect whether the image is clean or forged. In Ref.[17], the authors have stated that majority of the blind forgery detection techniques applied on the images are getting a higher error rate because of the fact these techniques sometimes fail due to the presence of noise in an image. So, they said that when the noise present in the image is used to detect the forgery along with the traditional methods they may yield better results. In Ref.[17], the image has first undergone wavelet transformation, then the image is divided into non overlapping blocks. The variance of noise is then estimated in each of the blocks. Finally, blocks are merged based on the variance values. The blocks with similar variance are merged together to form one group. Those blocks whose variance is largely deviated from the rest of the blocks are said to be forged. In Ref.[18], also the authors have divided the image into blocks and estimated the variance to detect the forged region but they have used a slightly different approach in order to highlight the forged region. The method uses two phases to correctly identify the forged region. In the first phase, after dividing the image into blocks and estimating the variance these blocks are first divided into categories using the k-means clustering algorithm. The blocks with different variance values are grouped into two clusters. The blocks of the cluster with less number of elements are assumed as forged. Then the image blocks which are clustered under the forged category undergo further investigation to accurately point out the forged region. This is the second phase of the method. In this phase, the images are again divided and noise estimation is done on these smaller blocks once again to exactly pinpoint the forged region.

3 AIMS AND OBJECTIVES

In this section, the aim and the objectives which were set in this research are stated. The research questions framed to reach these objectives have also been listed along with the motivation behind their formation.

3.1 Aim

The main aim of this thesis is to improve the detection rate of an algorithm proposed previously in the literature to detect splicing forgery in digital images. We also aim to analyze how the algorithms fares on low resolution images.

3.2 Objectives

The objectives set to accomplish these aims are as follows:

- To implement an algorithm for detecting splicing forgery in digital images.
- To examine the factors that may lead to the improvement of the detection rate of splicing forgery in images and their low resolution versions.
- To analyze the impact of image resolution, and the selected classifiers on the algorithms' performance.

3.3 Research Questions

The research questions which we tried to answer through this thesis are

RQ1. How can we improve the detection rate of splicing forgery in digital images?

Motivation: This research question was formulated in light of the widespread of digital images along with digital techniques used to manipulate them. Unethical manipulations of images is on the rise nowadays with the popularity of social media platforms (i.e., Facebook, Twitter, etc.). As stated earlier, images are being used to make important decisions [1]. Hence, improving the detection rate of forged images is very well desired.

RQ2. What is the impact of image resolution and the classifiers on the performance of the examined image tempering algorithm?

Motivation: Only the authors of Ref.[15][16] have considered the impact of their algorithm on low resolution images. Image resolution is an important factor as; in real life we generally do not have images which are of the same size or resolution. Moreover in Ref.[13][14], the authors have also stated that their algorithms do not perform well if the images are compressed. So, we have considered the resolution of images to check how well the algorithms fare on images of different sizes. The performance of the algorithm on different classifiers was also considered (i.e., we used SVM and simple Tree classifiers).

3.4 Contribution to the Research field

Through this thesis we are improving the detection rate of splicing forgery. Many authors have stated that improving the detection rate is a desired feature in the current research [19].

The rate of detection has been significantly improved through this research. We have found out that techniques such as the Discrete Wavelet transformation (DWT), YCbCr color transformation have improved the performance of our improved algorithm. Through this research we are also checking the impact of the image resolution and the selected classifiers on both of the algorithms (the original and our enhanced version). Impact of the image resolution on the algorithms is considered as some of the authors in the literature [15][16] have considered this aspect to analyze their algorithms. Other reason why image resolution is an important factor is that the images which we use in our daily life are not always of the same resolution. So, it is desirable to check how the algorithm works on images of different resolutions. We have also considered the impact of the classifiers and analyzed their performance on the considered algorithms through this thesis.

3.5 Limitations

In this thesis we are only focusing on the area of splicing forgery detection. This is because of the fact that the field of image forgery is very vast, therefore, only a part of it is considered in this research. The considered algorithm [5] was designed to detect if a given image is clean or forged even if it had undergone a counter-forensic measure of contrast enhancement. But, in this research we have not analyzed or conducted experiments on this aspect as we have limited our scope to splicing forgery. The considered algorithm labels images into two distinct classes; either clear or forged. The algorithm will not be able to localize the region where the image had been forged, that is another area of research.

4 APPROACH

In this section the approach used to conduct this thesis is documented. It describes about various accepts such as the tools, datasets used, how the models are created after feature extraction and how these models are tested. In this section the pseudo codes of algorithms used, *Algorithm 1* and *Algorithm 2*, are also presented.

4.1 Tools Used

The implementation and the experimentation of the algorithm had been carried out using Matlab®. We have used the Matlab® R2016a version that consists of the statistics and machine learning Toolbox [20] which is used for training the data using different classifiers.

4.2 Dataset Used

The datasets required for training and testing have been obtained from Ref.[10]. The images present in this dataset are color images. This dataset contains two different folders of images one with clean images and the other with images which are forged. The dataset contains a total of 183 clean images and 180 forged images. From the available data we have used half of the images for training the classifier and the rest for testing it. Hence, in the process 182 images out of which; 92 were clean and the rest spliced were used to train the model and the remaining 181, 91 clean and 90 forged were used for testing the model created during the training process.

4.3 Feature Extraction and Model Creation

We have considered the algorithm of Ref.[5] for our research. This algorithm was considered after searching online for a recently published algorithm of which its authors are willing to share the source code. We thought of considering an algorithm whose code was available due to the time factor, as it would take a lot of time if we were to implement the algorithm by ourselves and then analyze and improve upon it. From the available algorithms we have considered, this algorithm that we selected was one of the latest published algorithms which reflects the current state of the art in the area. An interesting attribute of this algorithm is that it is capable of detecting if any counter-forensic measures have been applied to hide the forgery process. Authors of another algorithm[13] which can deal with counter forensics have stated that their algorithm does not perform well if all the R, G, B channels have been enhanced equally. This area of counter-forensic attacks has not been studied in this research due to the time factor, but can be deferred to a future work.

In the original algorithm (referred to as *Algorithm 1* in this document), obtained from Ref.[21], the algorithm loads three images one clean image, one contrast enhanced image and one image on which counter-forensic techniques have been applied. The algorithm obtained was a partial implementation of the algorithm proposed in Ref.[5] as it does not implement the machine learning part. To the obtained algorithm of *Algorithm 1* additional features like capability of loading images automatically from a folder, facility to change the resolution of the image were added. The second feature to change the image resolution was embedded as the RQ2 deals with analyzing the performance of the algorithms on different resolutions. The algorithm has also been modified in such a way that it can even handle color images. This was done as the dataset we are using for testing and training the model consists of color images. We have considered a dataset of color images as we are considering the red, chromatic yellow, chromatic blue channels of the image in *Algorithm 2* (the modified version of *Algorithm 1*) and when we are comparing two algorithms they should be compared on the same datasets. We have also made sure that all the values of the extracted features, the class labels of whether images are considered clean or forged are stored in a table so that they can be used while training the model. We are training the model using clean and images which have undergone splicing forgery (forgery technique focusing in this research). The pseudo code of the algorithm derived from Ref.[5][21] is stated below. This algorithm is referred to as *Algorithm 1* in this thesis.

4.3.1 *Algorithm 1*

```
1. Create a table with image names
2. Authenticity = [ones([92 1]) ; zeros([90 1])];
3. Initializing detector parameters
4. Initializing arrays to store the features
5. For (all the images)
    a. I = Imread(image) // To read the image from the folder
    b. I = Imresize(I, n) // where I is the original image and n is the resampling ratio
       (e.g., 0.25)
    c. J = rgb2gray(I) // converting the RGB image to gray image to obtain co-
       occurrence matrix used in feature extraction
    d. Feature Extraction
    e. Store the values obtained after feature extraction
6. End For
7. Feed these values to the classifier for training
8. Save the generated model
```

Through this research we have tried to improve *Algorithm 1* such that the rate of detection of the forged images increases. The modified version of *Algorithm 1* is referred to as *Algorithm 2* in this document. In *Algorithm 2* each RGB image is initially converted into YCbCr color space and then each of the color channels (i.e., Y, Cb, Cr) undergoes single level 2D wavelet decomposition to give four images, namely, low pass approximation image, horizontal detail image, vertical detail image, diagonal detail image. In this research we have only considered the low pass approximation image to extract the features. This is because of the fact that only the low pass approximation image contains the whole image. The rest of the images contain either the horizontal, vertical or diagonal details [22]. In addition to the extracted features in *Algorithm 1* we have extracted additional features direct current value of the discrete cosine transformation and the entropy of both the image and the co-occurrence matrix of the image. The pseudocode for *Algorithm 2* is stated in 4.3.2 section.

The YCbCr color transformation derives three images from the original image. The Y, Cb, and Cr components. The Y component is the gray scale version of the image. The Cb, Cr are the chromatic blue and Chromatic red versions of the image, respectively. The YCbCr color transformation was considered as through this transformation the quality of the image is less affected by compression [23]. This is a desired feature as we are dealing with images of low-resolutions.

We have repeated the training and testing process on images of different resolutions. We have also built different models using different classifiers namely simple tree, linear SVM (see the Methodology section for more details).

4.3.2 *Algorithm 2* (Our enhanced version of *Algorithm 1*)

```
1. Create a table with image names
2. Authenticity = [ones([92 1]) ; zeros([90 1])];
3. Initializing detector parameters
4. Initializing arrays to store the features
5. For (all the images)
```

```
        a.  I = Imread(image) // To read the image from the folder
        b.  I = Imresize(I, n) // where I is the original image and n is the resampling ratio
            (e.g., 0.25)
        c.  Image_ycbcr = rgb2ycbcr(I) //RGB image is converted to YCbCr color
            space
        d.  For (each of the YCbCr channels)
              i.  Apply wavelet decomposition
             ii.  Consider only the low-pass approximation images
            iii.  For (each of the low-pass approximation image)
                    1.  Feature Extraction
                    2.  Store the values obtained after feature extraction
             iv.  End For
        e.  End For
    6.  End For
    7.  Feed these values to the classifier for training
    8.  Save the generated model
```

4.4 Testing new images on the constructed model

After generating the model by using the selected classifier the next step is to test and evaluate how the model performs. While testing how the model performs on the test dataset, initially all the steps used to train the model are followed except feeding the data to the classifier and generating a model. Here instead of feeding the data and training the model we use the saved model to predict the class of the image i.e. whether it is clean or forged. After predicting the classes of the test dataset they are compared with the actual classes to check the accuracy of the algorithm. Both the models built using algorithms *Algorithm 1* and *Algorithm 2* are tested using the test datasets and their accuracy is calculated to check how well an algorithm works.

5 METHODOLOGY

The research methods used in the process of our thesis are implementation and experimentation. We have examined the different possible types of research methods using Ref.[24]. Research methods such as interviews, surveys, case-studies are not apt for this research. This is because of the reason that through interviews and surveys we get the opinion of people. Through case-studies we get to observe and study about an already existing situation or a group. All these research methods are not suitable for this research as the algorithms cannot be improved or tested based on human opinion. In order to improve the detection rate for the first RQ, we have to modify the existing algorithm which can only be done using the implementation method. For the second RQ we have to show how the dependent variables change with respect to the independent variables. In such type of cases experiments are the most suited type of the research methods. So, we have chosen this as the research method to answer this RQ.

Hence, implementation is used to answer RQ1. Whereas we have conducted experiments in order to analyze how the implemented algorithm works on different resolutions of the image. We have considered simple tree and linear SVM classifiers to conduct the experiments on images of different resolutions. SVM was considered as the authors of Ref. [5] have used this for classification. On *Algorithm 2*, 5-fold cross validation was conducted using different classifiers on the training dataset to find out the estimated accuracy of the algorithm. Simple tree was considered from the classifiers after it gave the highest estimated accuracy. Later we have even tested the models on a new test dataset so as to check the accuracy and generalizability. More detailed description along with the results of 5-fold cross validation are presented in 6.2 sub-section of the Results section.

5.1 Operation

RQ1: The research method used to answer the first RQ is implementation. We have considered *Algorithm 1* proposed by the authors of Ref.[5]. We found the source code for partial implementation of this algorithm from Ref.[21]. The available source code did not contain the machine learning part and also was working on only gray images. We have modified the code such that it can be used on color images as well. The source code is also modified in such a way that the algorithm automatically takes the images from a folder converts it into a gray image and then the feature extraction was carried out.

In our enhanced *Algorithm 2* we have first converted the color image into a YCbCr color space. Next each of the gray, chromatic blue, chromatic red images are considered separately and then the single level discrete wavelet decomposition is applied on each of the images. After applying the single level discrete wavelet decomposition[22] on the image we have considered only the low-pass approximation image from the available four bands to proceed with feature extraction.

Apart from the co-occurrence matrix used in *Algorithm 1* we have also extracted features like entropy of the image, entropy of the obtained co-occurrence matrix, the direct current value obtained after using the discrete cosine transformation. All these features are extracted from each of the three images obtained from the YCbCr transformed image unlike the actual algorithm which directly considers only the gray image. As a result, we were able to extract a total of 771 features when compared to the 254 features collected by the authors of the paper.

RQ2: To answer the second research question we have conducted experiments to check how the algorithms work on images of different resolutions and how the considered classifiers perform on these images of different resolution. All the experiments were conducted on the test dataset. The experiments were conducted using two classifiers, Linear SVM and Simple

Tree. Two sets of experiments were conducted, in the first one, the model was trained on the original images, and subsequently tested on images of different resolutions. In the second set of experiments the model was trained and tested on images of the same resolution. All the conducted experiments are listed in the section 7.3 in the form of cases for easy reference during analysis.

Three types of resolutions were considered to conduct the experiments, original images and the resized images. We have resized the available images to half, quarter and applied *Algorithm 1* and *Algorithm 2* on these images. Then we have calculated the accuracy, AUC of the algorithm in each of these cases and analyzed them.

5.2 Independent and Dependent Variables in Experiments

Independent variables are those which are changed in the experiment and these variables are not dependent on the other variables used in the experiment [25]. Dependent variables are those variables which change based on the change of the independent variables [25]. In the conducted experiments, the image resolution and the chosen classifiers are the independent variables. The values of the accuracy and the AUC parameter are the dependent variables as these values change based on the type of classifier and image resolution. Image features are also considered to be dependent variables on the level of resolution.

5.3 Validity Threats

In this section we highlight the potential validity threats that may occur while conducting the thesis and the actions taken to reduce these threats.

5.3.1 Internal Validity Threats

Internal validity makes sure that the research is conducted correctly without any errors. It also sees that the experimental setup does not affect the results[26][27]. As we have chosen experiments to answer our RQ2 there is a possibility that the results obtained such as the accuracy of the classifier may not be accurate. We made sure that such event did not happen by cross-checking the experiments conducted. For the test data the accuracy was calculated manually by checking out the false positives and true negatives, so there is a probability that the humans may make mistakes but we made sure that this did not happen by checking out the false positives and true negatives twice. The selected dataset for training and testing may have undergone selection bias and may vary depending on the experimenter.

5.3.2 External Validity Threats

External validity threats are the threats through which the generalization of the conducted experiment may be limited [26] [27]. In order to check the generalization of the experiment, the datasets considered for training and testing should be different. We have also checked the algorithms on images of three different resolutions to check how the algorithms perform on images of different resolutions. It is observed that there is only a slight variation in the performance of the algorithms on images of different resolutions. But, generalization of the proposed method to other types of forgery may require re-training and an adequate training dataset. Therefore, the reported accuracy rates are limited to the dataset we examined and to the type of forgery we examined.

5.3.3 Construct Validity Threats

Through Construct validity we can make sure that our study has correctly focused on the research area we are dealing with and the obtained results are not deviated [27]. In this research to analyze how the algorithm works on images of different resolutions, we have considered images of three types instead of just basing our results on just two resolutions. The results are based on the experimentation on these resolutions.

6 RESULTS

In this section, the obtained results after implementation and experimentation are documented. The implementation sub-section states about the results obtained after the implementation of *Algorithm 2* and has results that can be used to answer RQ1. The Experimentation sub-section documents the results of the experiments conducted. These results are obtained to answer RQ2.

6.1 Implementation

We have been successful in implementing and improving the algorithm considered for the images which have undergone splicing forgery. The authors of the original paper, on which this thesis is based on, have stated that they have used SVM algorithm to classify the images but we have tested the algorithm using both linear SVM and a simple tree. As stated in the methodology in the previous section we have extracted a total of 771 features as opposed to the 254 features used in the original algorithm. Second-order statistics were used as features in comparison with the first order statistics like histogram as the features extracted by the first order statistics like histogram could be easily manipulated using counter-forensic methods [5].

Algorithm 2 performs better than *Algorithm 1* in classifying the images as clean or tampered. This is the case with all the considered classifiers. By seeing Figure 6.1 this can be clearly visualized. The figure shows the accuracy of classifiers simple tree and linear SVM on the test dataset using both the algorithms *Algorithm 1* and *Algorithm 2*. But the time taken to extract the features in *Algorithm 1* is very less as compared to *Algorithm 2*. It takes about 20.927 seconds to extract the features required to train the model in the case of *Algorithm 1* and about 78.469 seconds for *Algorithm 2*. This is due to the large difference between the number of features extracted in *Algorithm 1* and *Algorithm 2*. A more detailed analysis on the performance of the two algorithms is described in the next section of analysis and discussion.

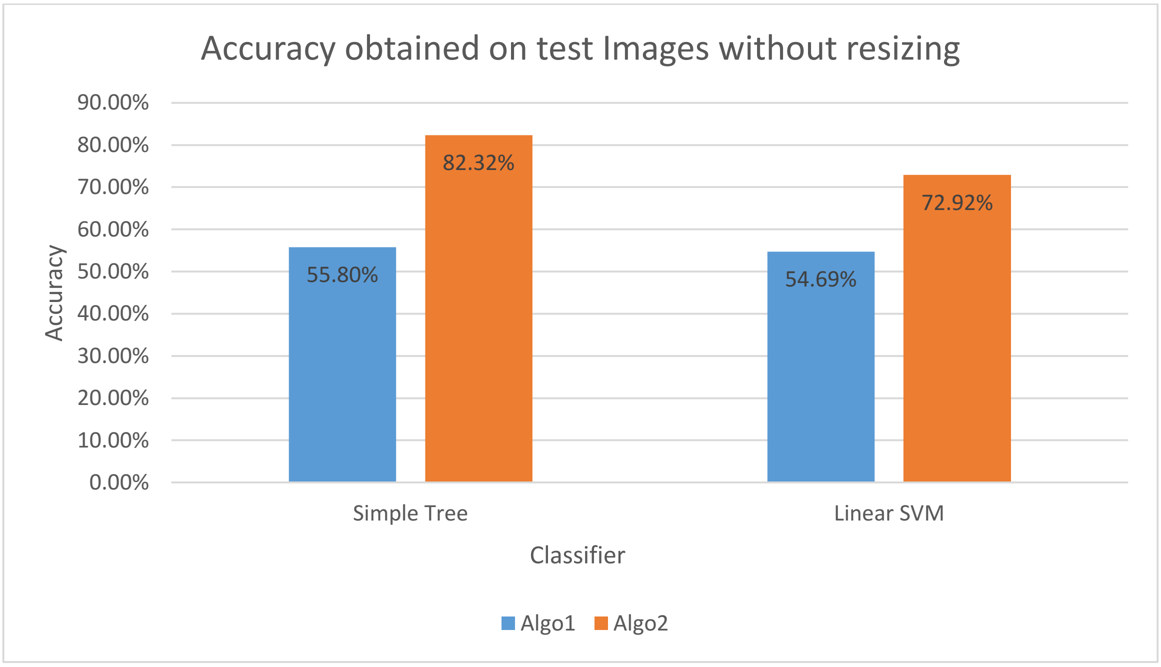


Figure 6.1: Classification comparisons between the two algorithms: Graph showing the Accuracy obtained on the test data of the images.

6.2 Experimentation

For conducting the experiments, we have considered both of the algorithms *Algorithm 1* and *Algorithm 2*. The training dataset was considered and two models were built using *Algorithm 1* and *Algorithm 2*. We have tried different types of classifiers available in the Matlab® Toolbox and considered the top classifier from the available set to perform our experiments apart from the SVM which was considered by the authors of Ref.[5] which proposed the forgery detection algorithm reported in this thesis. The top classifier was selected based on the predicted accuracy of the classifier which is calculated by the classifier using 5-fold cross validation. We have considered different types of classifiers to trace their impact on the performance of the different Algorithms (RQ2).

The predicted accuracy of the classifiers obtained after conducting the 5-fold cross validation on the training dataset on *Algorithm 2* are tabulated in table 6.1. From Table 6.1 it can be observed that the Simple Tree has the highest accuracy with 75.3%. Hence, it was considered along with SVM for further experimentation.

Classifier	Accuracy (%)
Complex Tree	67.0
Medium Tree	67.0
Simple Tree	75.3
Linear Discriminant	63.7
Quadratic Discriminant	64.3
Logistic Regression	51.6
Fine KNN	43.4
Medium KNN	52.2
Coarse KNN	54.9
Cosine KNN	59.9
Cubic KNN	53.3
Weighted KNN	50.0
Boosted Trees	50.5
Bagged Trees	73.6
Subspace Discriminant	47.8
Subspace KNN	69.8
RUSBoosted Trees	58.2

Table 6.1: Classifiers and their accuracy obtained after 5-fold cross validation on the training dataset.

After the models have been built using the selected classifiers, both the models are tested on images of different resolution. Then the test dataset was used to test the performance of the algorithms. The performance measures considered are the accuracy of the algorithm, AUC as in our thesis we are dealing with the detection of image tampering.

To answer the second research question which is to check the impact of the image resolution on the considered image tampering algorithms initially only the test dataset is converted into two other resolutions and the model was tested on these image resolutions also. So each model or the algorithm's performance was measured on three types of datasets of different resolutions. One was the original images, in the second and third datasets of the images the images were resized by half, and quarter respectively. Then the count of the number of images which have been correctly classified has been noted down. The accuracy of the algorithm on these different datasets was calculated using equation 1 (see sub-section 1.1.3.1). The total number of instances used to calculate the accuracy is equal to the number of images in the test dataset which is equal to 181 in this case.

The results of the experiments on *Algorithm 1* and *Algorithm 2* when trained on images without resizing and tested on images of different resolutions are tabulated in Table 6.1 and Table 6.2.

	Algorithm 1		***Algorithm 2***	
Image Resolution	Correctly Classified Instances	Accuracy	Correctly Classified Instances	Accuracy
Original Images (n= 181)	99	54.69%	132	72.93%
Resize (0.5)	103	56.90%	116	64.08%
Resize (0.25)	103	56.90%	107	59.11%

Table 6.2: Performance of *Algorithm 1* and *Algorithm 2* using Linear SVM when trained on images without resizing.

	Algorithm 1		***Algorithm 2***	
Image Resolution	Correctly Classified Instances	Accuracy	Correctly Classified Instances	Accuracy
Original Images (n= 181)	101	55.80%	149	82.32%
Resize (0.5)	95	52.48%	122	67.40%
Resize (0.25)	96	53.03%	128	70.71%

Table 6.3: Performance of *Algorithm 1* and *Algorithm 2* using Simple tree when trained on images without resizing.

The Figure 6.2 graphically represents how the accuracy of both the algorithms vary with each other as well as the variation of the accuracy with respect to the resolution of the images is also shown. It can be clearly observed that *Algorithm 2* performs better than *Algorithm 1* in each of the cases. The impact of the image resolution on the algorithm's performance is analyzed in the analysis and discussion section.

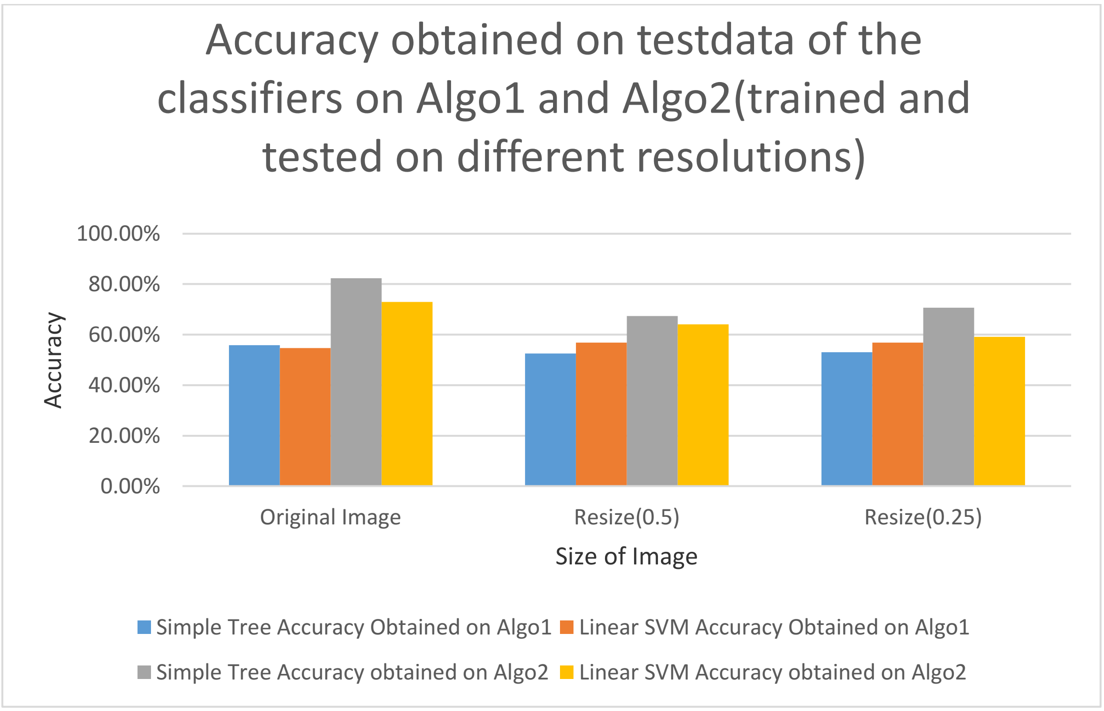


Figure 6.2: Accuracy of the classifiers on different image resolutions on *Algorithm 1* and *Algorithm 2* when trained on the original images and tested on images of different resolutions.

After conducting the experiments by only changing the resolution of the images in the test dataset, experiments were conducted by training and testing the model with images of similar resolutions. These results are tabulated in table 6.4 and can be visualized through figure 6.3.

Classifier used	***Algorithm 1***		***Algorithm 2***	
	Correctly Classified Instances	Accuracy	Correctly Classified Instances	Accuracy
For Original images				
Simple Tree	101	55.8%	149	82.3%
Linear SVM	99	54.69%	132	72.93%
Image size is reduced to half (resize (0.5))				
Simple Tree	94	51.93%	137	75.6%
Linear SVM	101	55.8%	136	75.14%
Image size is reduced to quarter (resize (0.25))				
Simple Tree	95	52.49%	144	79.5%
Linear SVM	92	50.83%	129	71.27%

Table 6.4: Performance of *Algorithm 1* and *Algorithm 2* when trained and tested on images of similar resolution.

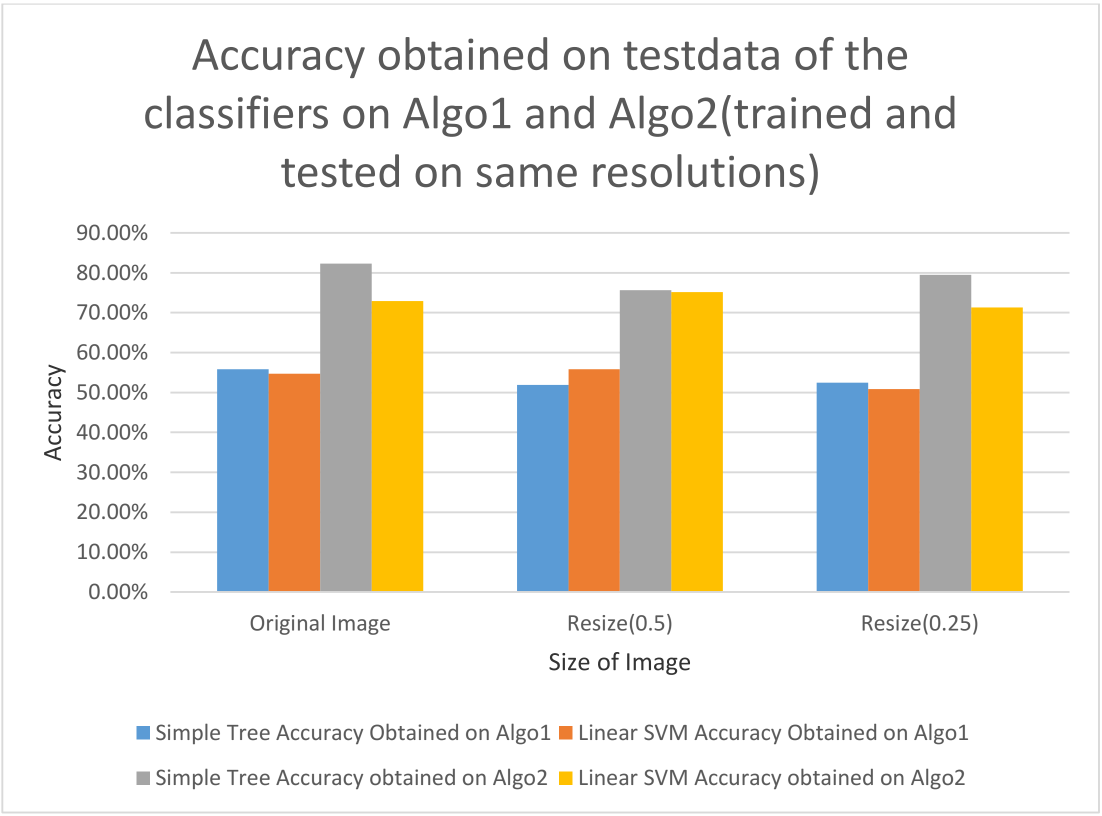


Figure 6.3: Accuracy of the classifiers on different image resolutions on *Algorithm 1* and *Algorithm 2* when trained and tested on images of the same resolution.

7 Analysis and Discussion

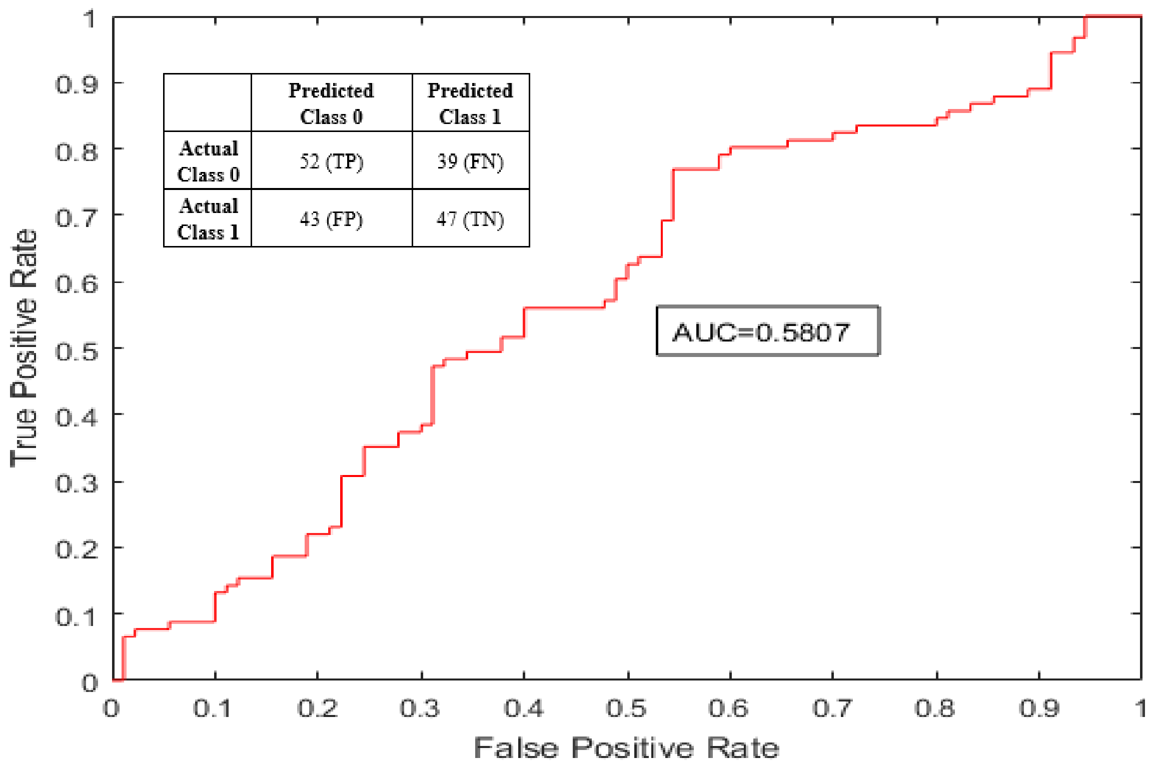
In this section the results are analyzed, discussed and the research questions are answered. To start analyzing the results obtained in the experiments, the confusion matrices of each case are derived and the receiver operating characteristic (ROC) curves are plotted. The ROC curves are constructed by choosing the threshold values of the probability based on which the instances are classified as either clean or forged in each case. If the obtained probability of the instance is below the chosen threshold, then it is classified as forged else it is clean. Then in each case the 1-specificity or the false positive rate (FPR) and the sensitivity or the true positive rate (TPR) are calculated and the receiver operating characteristic (ROC) curves are drawn. The ROC curve has FPR on the x- axis and the TPR on the y-axis. The ROC curves are displayed through the figures 7.1 to 7.20. After drawing the ROC curves, the area under the curve (AUC), is computed for each ROC curve.

7.1 Algorithm 1

The results of *Algorithm 1* are analyzed in this sub-section. This section is further divided into two sections, one where the images are trained on original image resolution and are tested on different considered resolutions. The second section deals with experiments where the training and testing are done on the same image resolution.

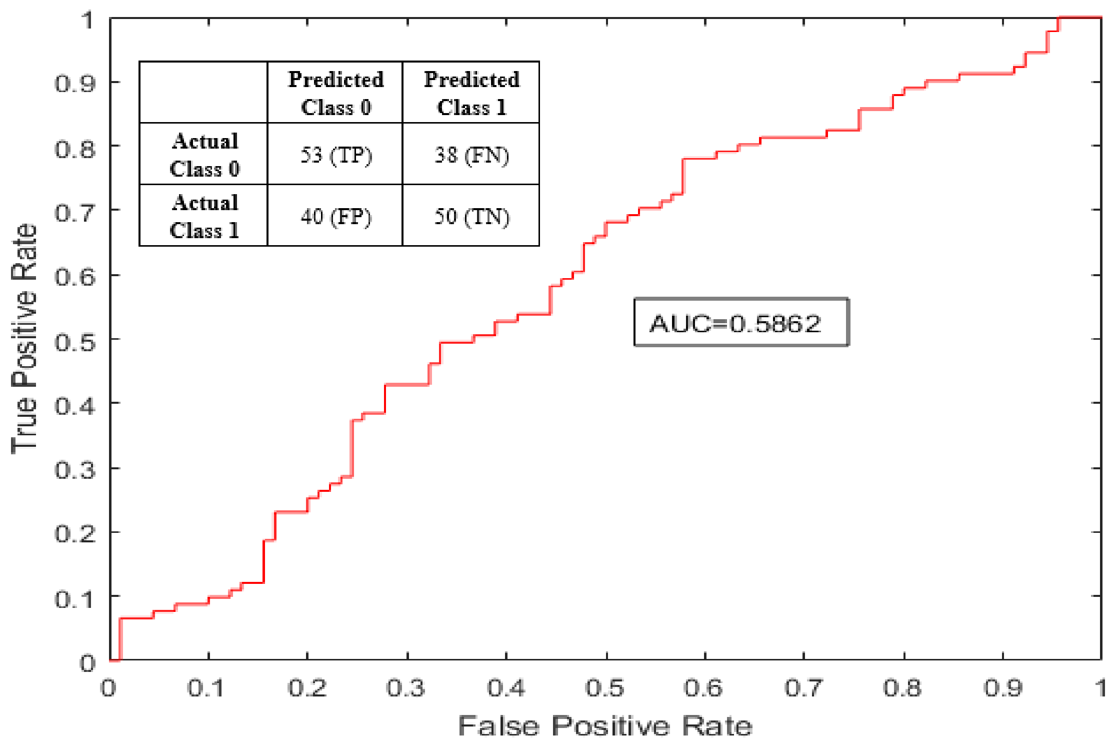
7.1.1 Training done on original images and tested on different image resolutions

Figures 7.1 to 7.6 show the performance of *Algorithm 1* on both classifiers using the AUC parameter. Table 7.1 tabulates the parameters: sensitivity, precision, and F-measure. Detailed description of these parameters and how they are calculated is stated in the background section.



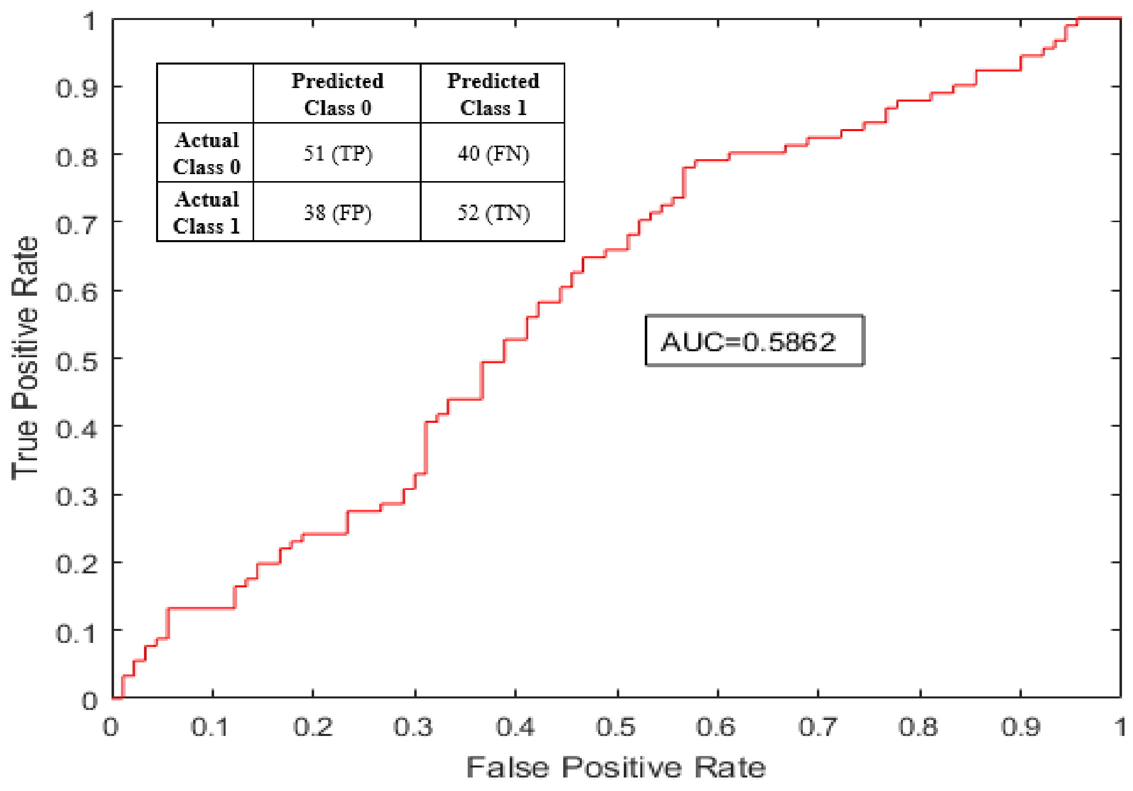
	Predicted Class 0	Predicted Class 1
Actual Class 0	52 (TP)	39 (FN)
Actual Class 1	43 (FP)	47 (TN)

Figure 7.1: ROC curve and Confusion matrix of *Algorithm 1* on Linear SVM.



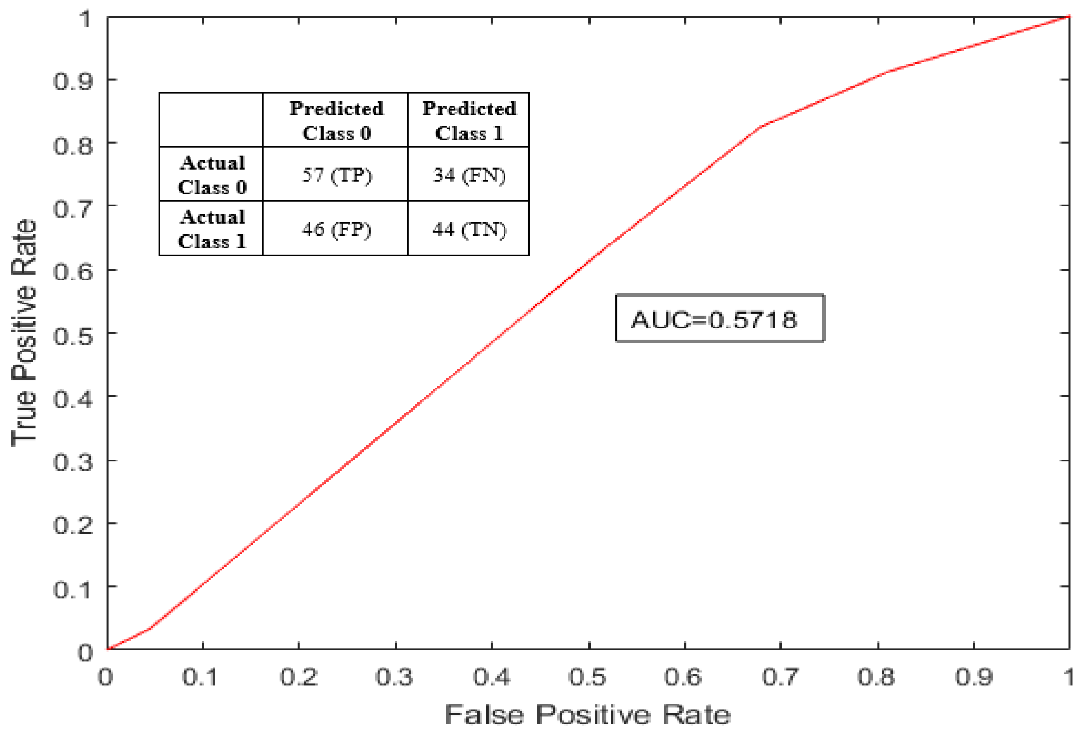
	Predicted Class 0	Predicted Class 1
Actual Class 0	53 (TP)	38 (FN)
Actual Class 1	40 (FP)	50 (TN)

Figure 7.2: ROC curve and confusion matrix of *Algorithm 1* on Linear SVM with image resolution of 0.5.



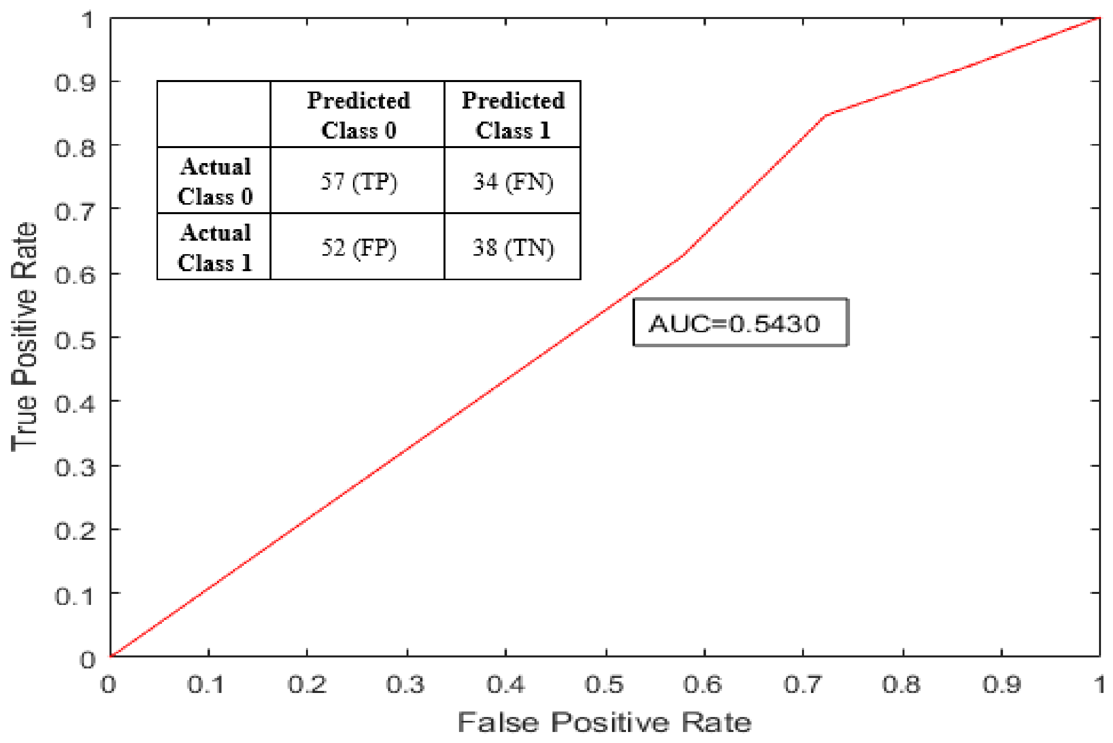
	Predicted Class 0	Predicted Class 1
Actual Class 0	51 (TP)	40 (FN)
Actual Class 1	38 (FP)	52 (TN)

Figure 7.3: ROC curve and confusion matrix of *Algorithm 1* on Linear SVM with image resolution of 0.25.



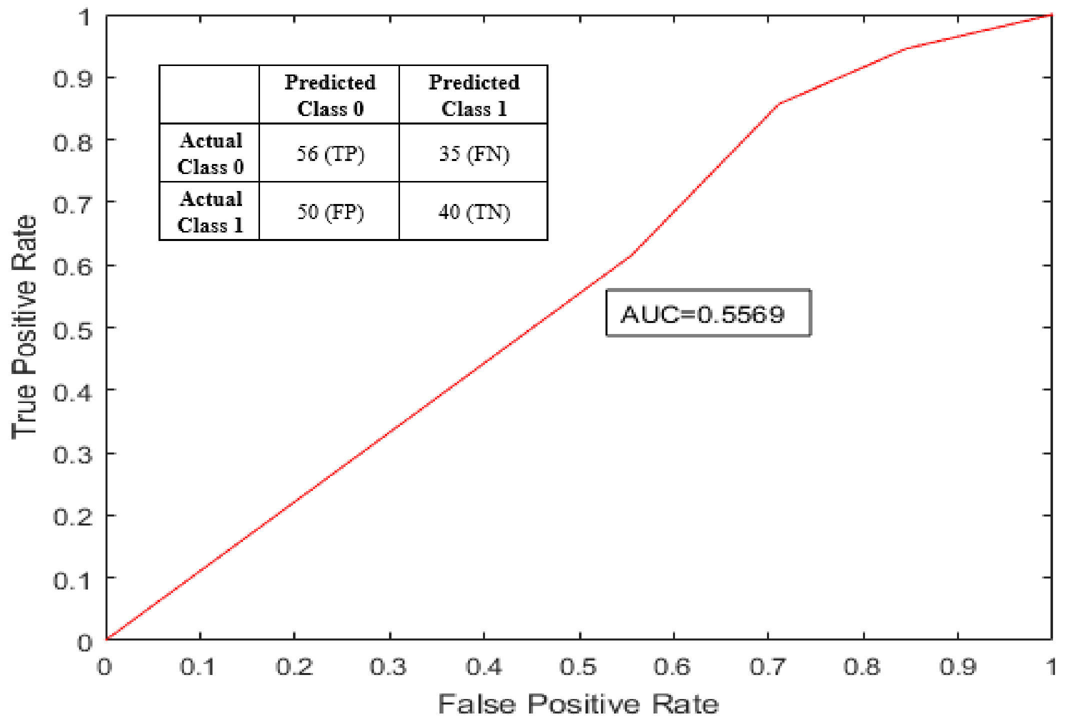
	Predicted Class 0	Predicted Class 1
Actual Class 0	57 (TP)	34 (FN)
Actual Class 1	46 (FP)	44 (TN)

Figure 7.4: ROC curve and confusion matrix of *Algorithm 1* on Simple Tree.



	Predicted Class 0	Predicted Class 1
Actual Class 0	57 (TP)	34 (FN)
Actual Class 1	52 (FP)	38 (TN)

Figure 7.5: ROC curve and confusion matrix of *Algorithm 1* on Simple Tree with image resolution of 0.5.



	Predicted Class 0	Predicted Class 1
Actual Class 0	56 (TP)	35 (FN)
Actual Class 1	50 (FP)	40 (TN)

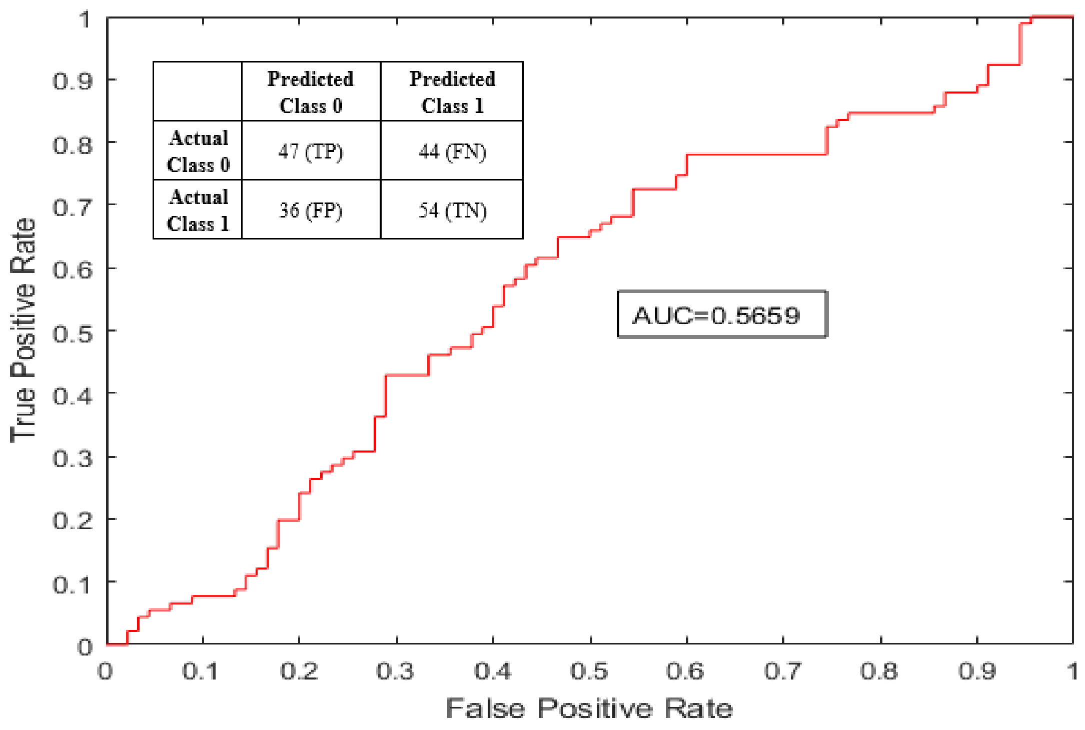
Figure 7.6: ROC curve and confusion matrix of *Algorithm 1* on Simple Tree with image resolution of 0.25.

	Linear SVM			**Simple Tree**		
Image Resolution	Sensitivity	Precision	F- Measure	Sensitivity	Precision	F-measure
Original Images (n= 181)	0.571	0.547	0.559	0.626	0.553	0.587
Resize (0.5)	0.582	0.570	0.477	0.626	0.523	0.570
Resize (0.25)	0.560	0.573	0.566	0.615	0.528	0.568

Table 7.1: Performance measures of *Algorithm 1* when training is done on original images and tested on images of different resolution.

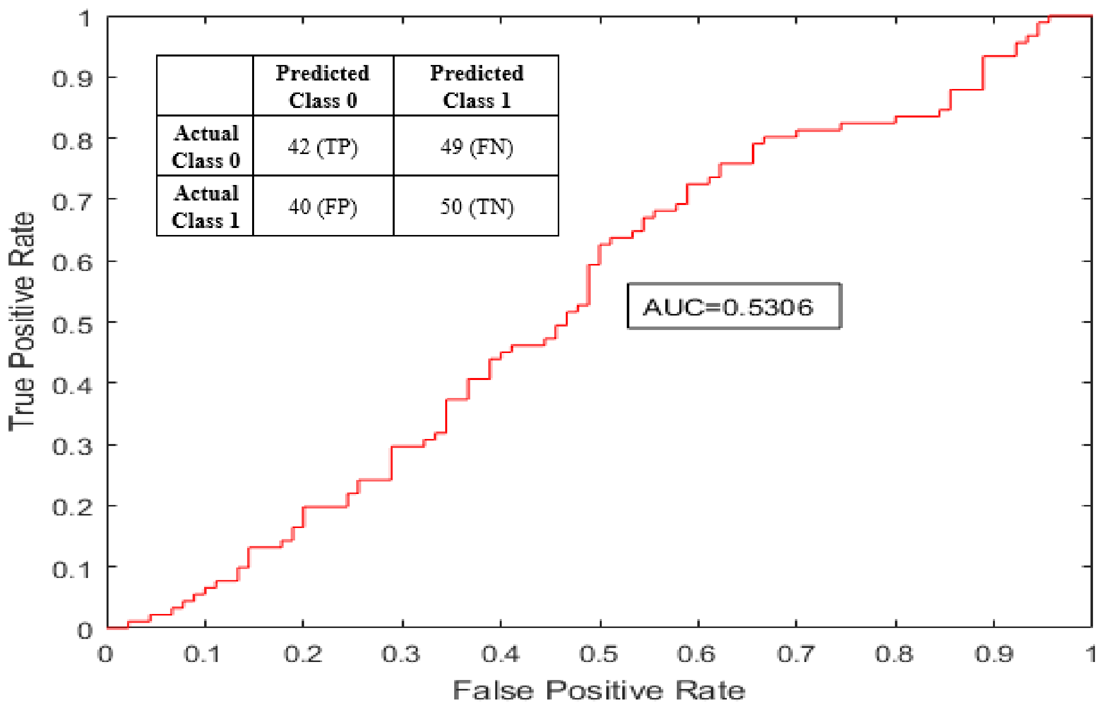
7.1.2 Training and testing done on images of the same resolution

In this section the ROC curves for the cases where the training and testing dataset are carried out on images of the same resolution. The performance of the algorithms on the original images is not included in this sub-section since it is already discussed in the sub-section 7.1.1.



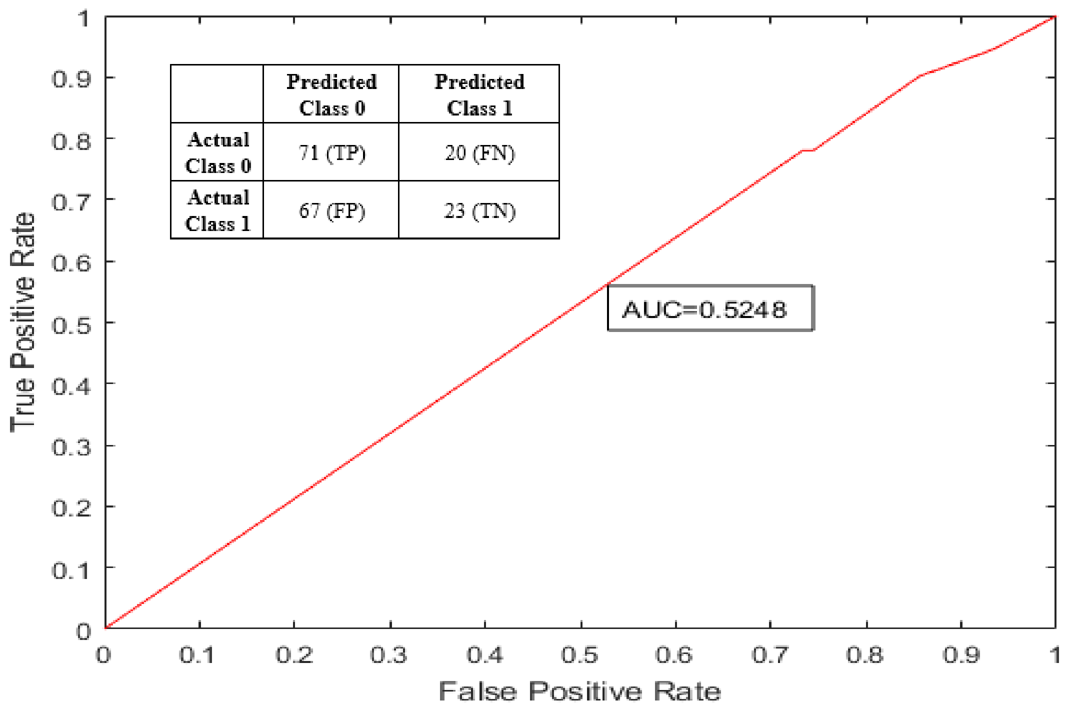
	Predicted Class 0	Predicted Class 1
Actual Class 0	47 (TP)	44 (FN)
Actual Class 1	36 (FP)	54 (TN)

Figure 7.7: ROC curve and confusion matrix of *Algorithm 1* on Linear SVM with image resolution of 0.5.



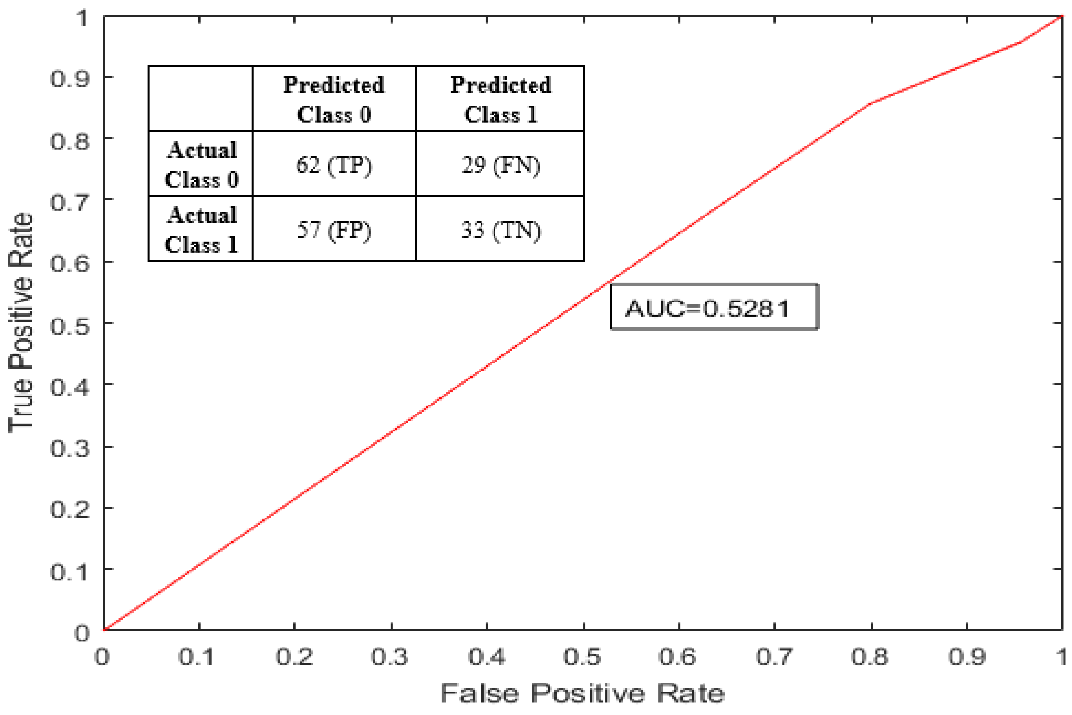
	Predicted Class 0	Predicted Class 1
Actual Class 0	42 (TP)	49 (FN)
Actual Class 1	40 (FP)	50 (TN)

Figure 7.8: ROC curve and confusion matrix of *Algorithm 1* on Linear SVM with image resolution of 0.25.



	Predicted Class 0	Predicted Class 1
Actual Class 0	71 (TP)	20 (FN)
Actual Class 1	67 (FP)	23 (TN)

Figure 7.9: ROC curve and confusion matrix of *Algorithm 1* on Simple Tree with image resolution of 0.5.



	Predicted Class 0	Predicted Class 1
Actual Class 0	62 (TP)	29 (FN)
Actual Class 1	57 (FP)	33 (TN)

Figure 7.10: ROC curve and confusion matrix of *Algorithm 1* on Simple Tree with image resolution of 0.25.

	Linear SVM			**Simple Tree**		
Image Resolution	Sensitivity	Precision	F- Measure	Sensitivity	Precision	F-measure
Resize (0.5)	0.516	0.566	0.540	0.780	0.514	0.620

Resize (0.25)	0.462	0.512	0.486	0.681	0.521	0.590

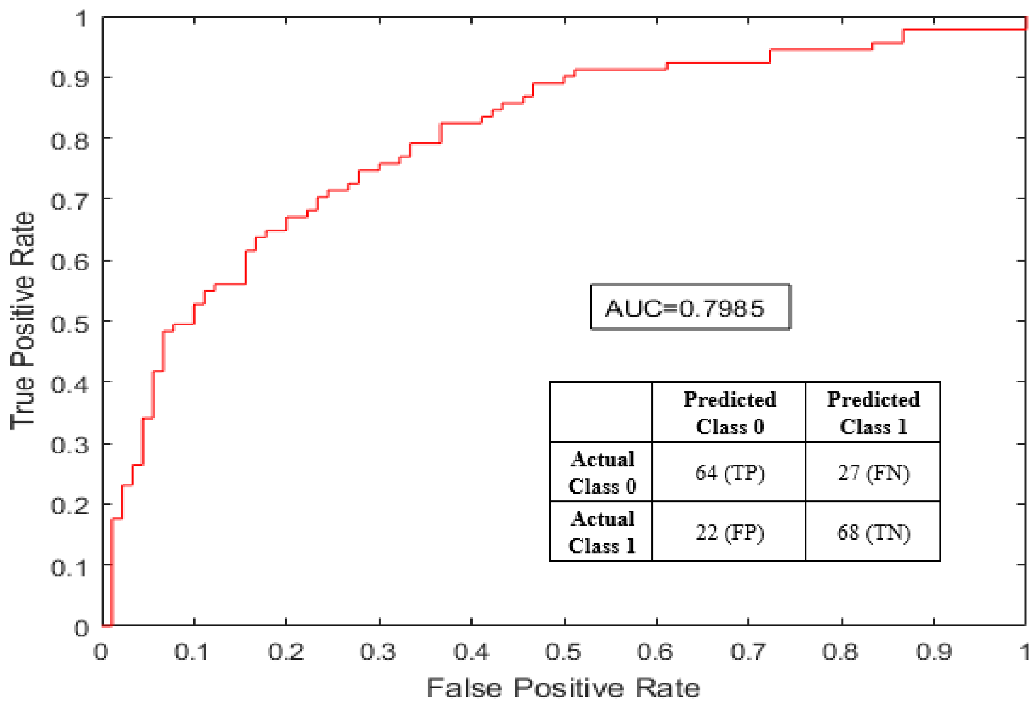
Table 7.2: Performance measures of *Algorithm 1* when training and testing is done on images of different resolution.

7.2 Algorithm 2

This section deals with the analysis of *Algorithm 2*; our enhanced version of *Algorithm 1*. As stated earlier, the experiments are conducted either by training and testing on the images of the same resolution or by training the model on original images and testing the model generated on images of different resolutions.

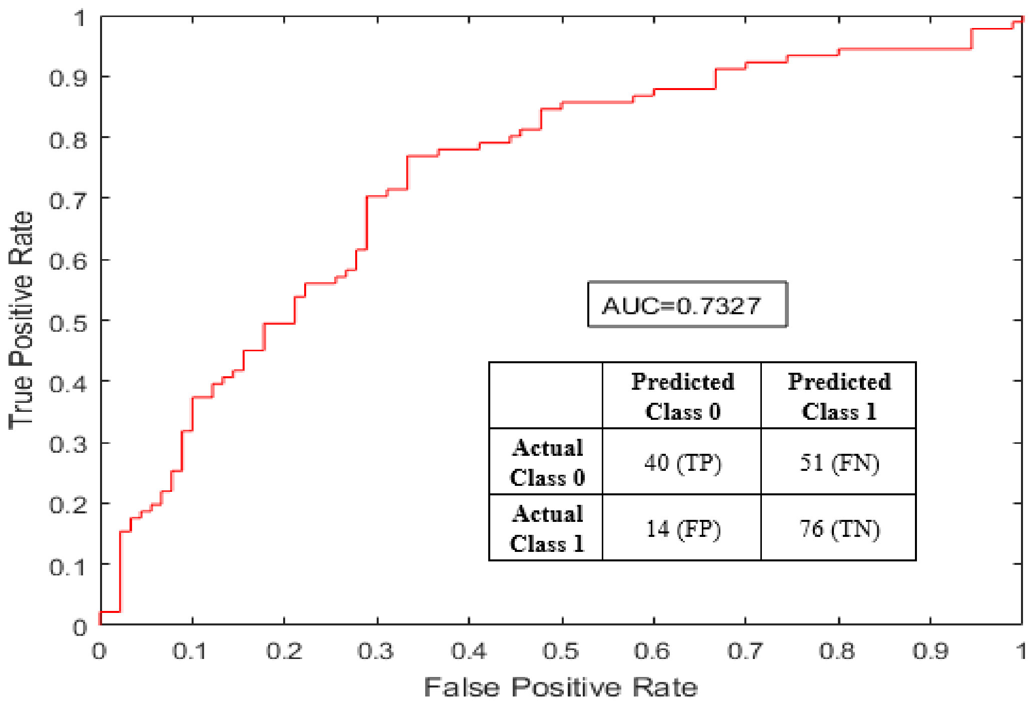
7.2.1 Training done on original images and tested on different image resolutions

Figures 7.11 to 7.16 show the performance of *Algorithm 2* on both of the classifiers using the ROC plot. Table 7.3 tabulates the parameters: sensitivity, precision, and F-measure. Detailed description of these parameters and how they are calculated is stated in the background section.



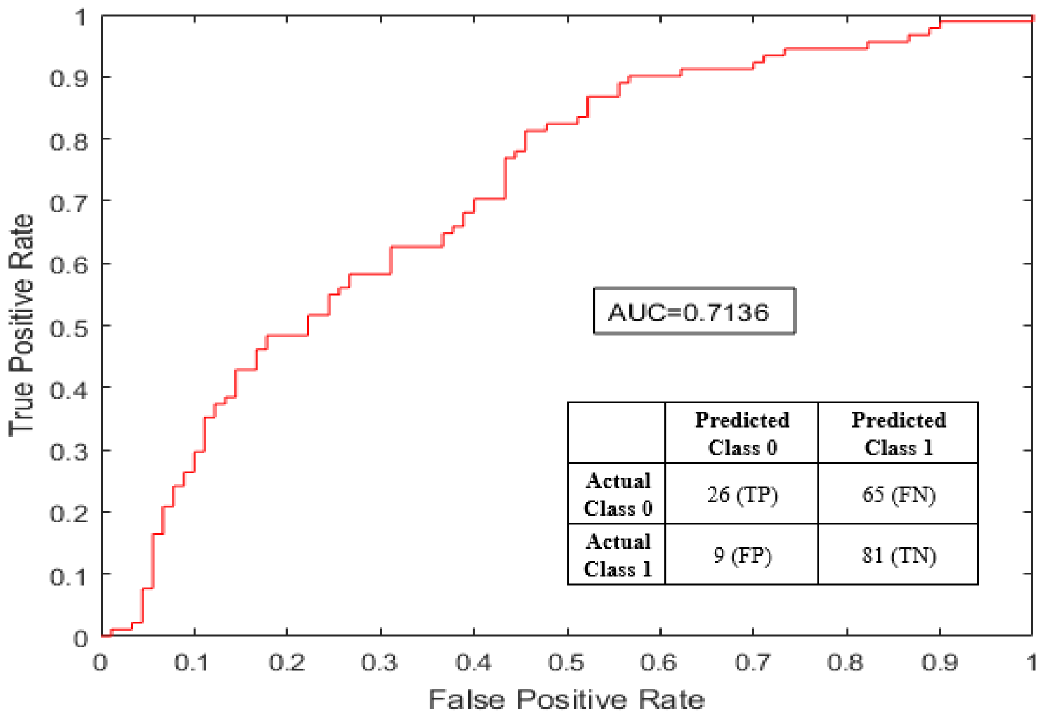
	Predicted Class 0	Predicted Class 1
Actual Class 0	64 (TP)	27 (FN)
Actual Class 1	22 (FP)	68 (TN)

Figure 7.11: ROC curve and confusion matrix of *Algorithm 2* on Linear SVM.



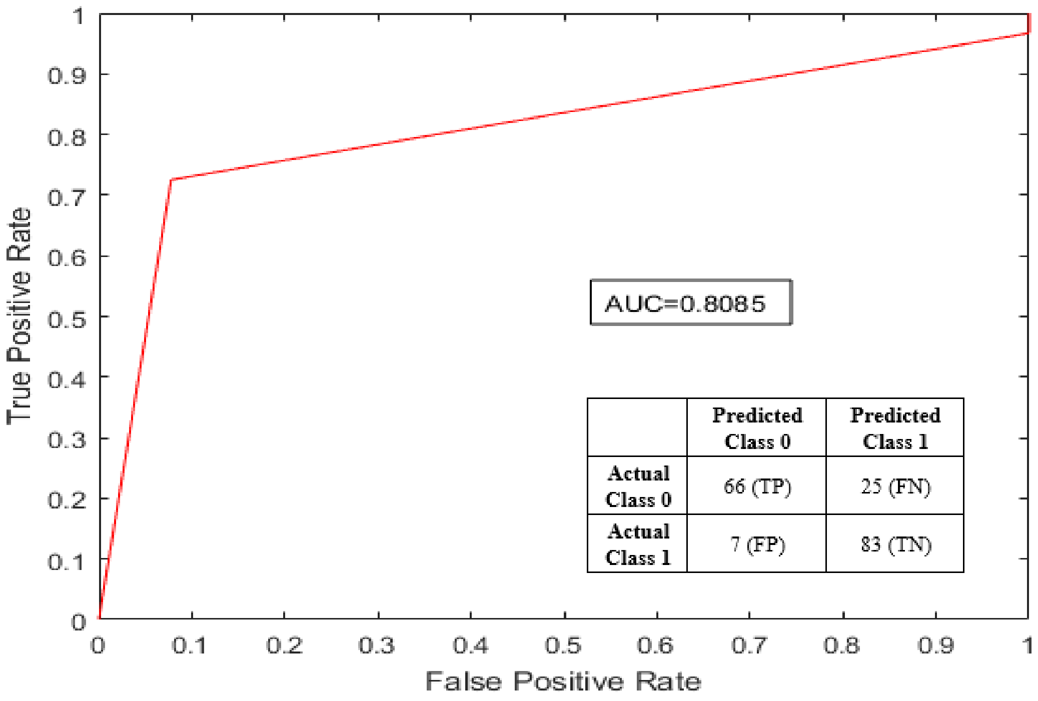
	Predicted Class 0	Predicted Class 1
Actual Class 0	40 (TP)	51 (FN)
Actual Class 1	14 (FP)	76 (TN)

Figure 7.12: ROC curve and confusion matrix of *Algorithm 2* on Linear SVM with image resolution of 0.5.



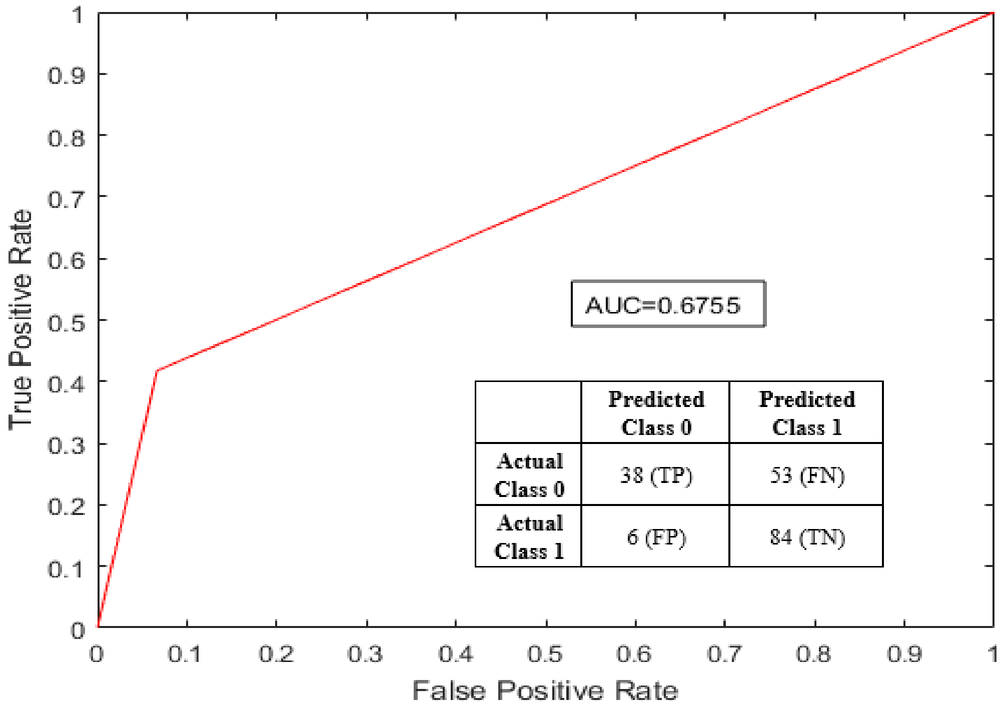
	Predicted Class 0	Predicted Class 1
Actual Class 0	26 (TP)	65 (FN)
Actual Class 1	9 (FP)	81 (TN)

Figure 7.13: ROC curve and confusion matrix of *Algorithm 2* on Linear SVM with image resolution of 0.25.



	Predicted Class 0	Predicted Class 1
Actual Class 0	66 (TP)	25 (FN)
Actual Class 1	7 (FP)	83 (TN)

Figure 7.14: ROC curve and confusion matrix of *Algorithm 2* on Simple Tree.



	Predicted Class 0	Predicted Class 1
Actual Class 0	38 (TP)	53 (FN)
Actual Class 1	6 (FP)	84 (TN)

Figure 7.15: ROC curve and confusion matrix of *Algorithm 2* on Simple Tree with image resolution of 0.5.

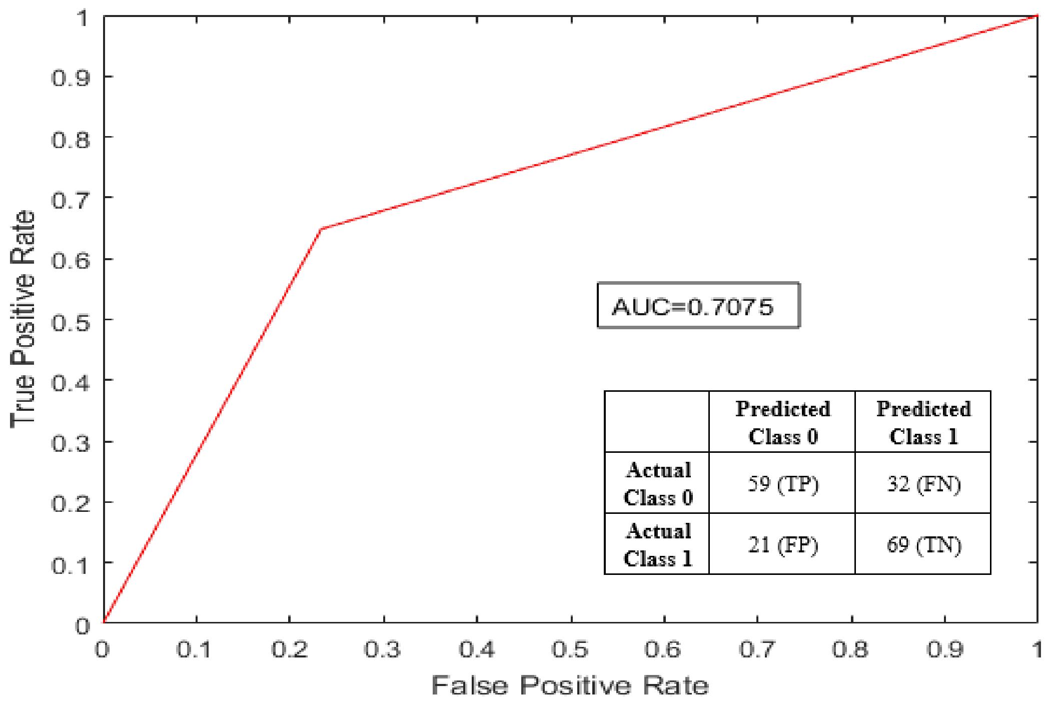


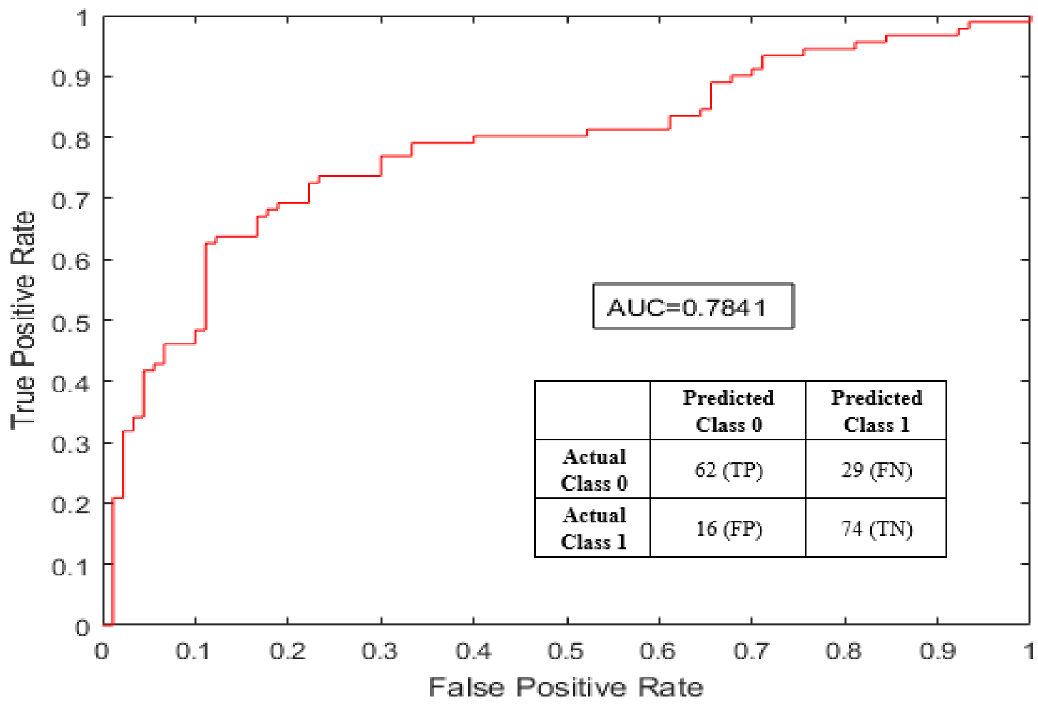
Figure 7.16: ROC curve and confusion matrix of *Algorithm 2* on Simple Tree with image resolution of 0.25.

	Linear SVM			**Simple Tree**		
Image Resolution	Sensitivity (TPR)	Precision	F- Measure	Sensitivity	Precision	F-measure
Original Images (n= 181)	0.703	0.744	0.723	0.725	0.904	0.805
Resize (0.5)	0.440	0.741	0.552	0.418	0.864	0.563
Resize (0.25)	0.286	0.743	0.413	0.648	0.738	0.690

Table 7.3: Performance measures of *Algorithm 2* when training is done on original images and tested on images of different resolution.

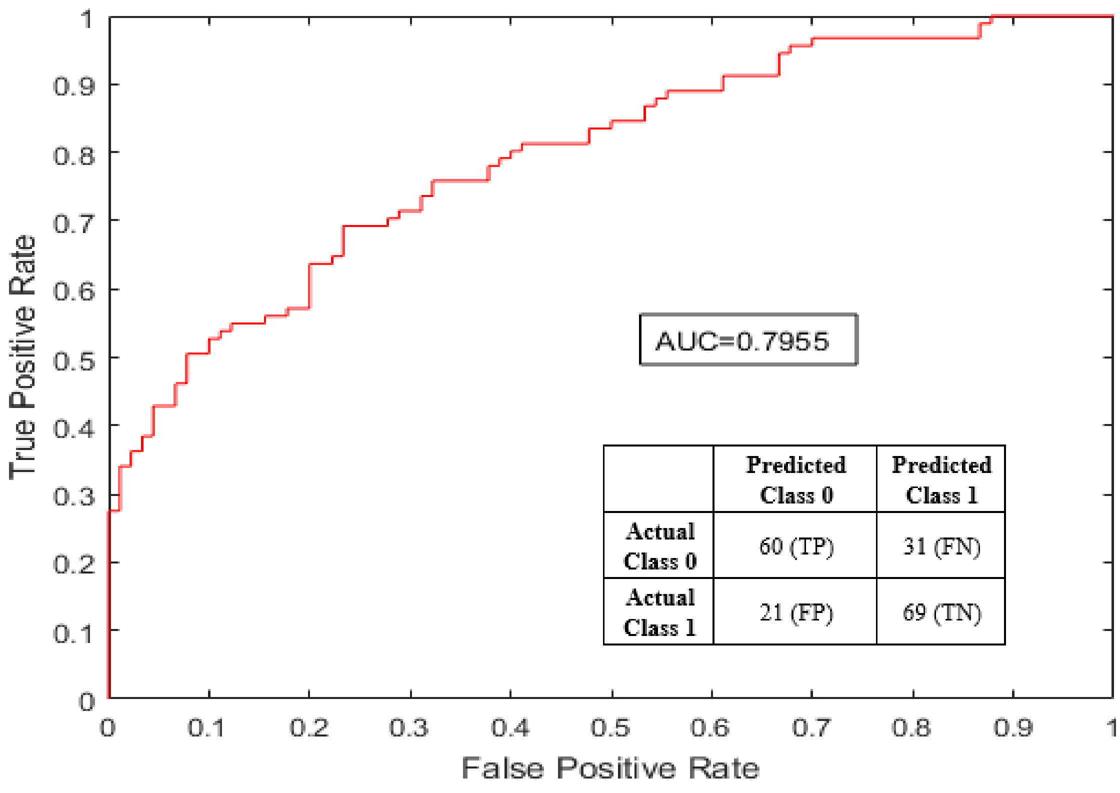
7.2.2 Training and testing done on images of the same resolution

Figures 7.17 to 7.20 show the performance of *Algorithm 2* on both of the classifiers when trained and tested on the same image resolution using the AUC plot. Table 7.4 tabulates the parameters: sensitivity, precision, and F-measure. Detailed description of these parameters and how they are calculated is stated in the background section.



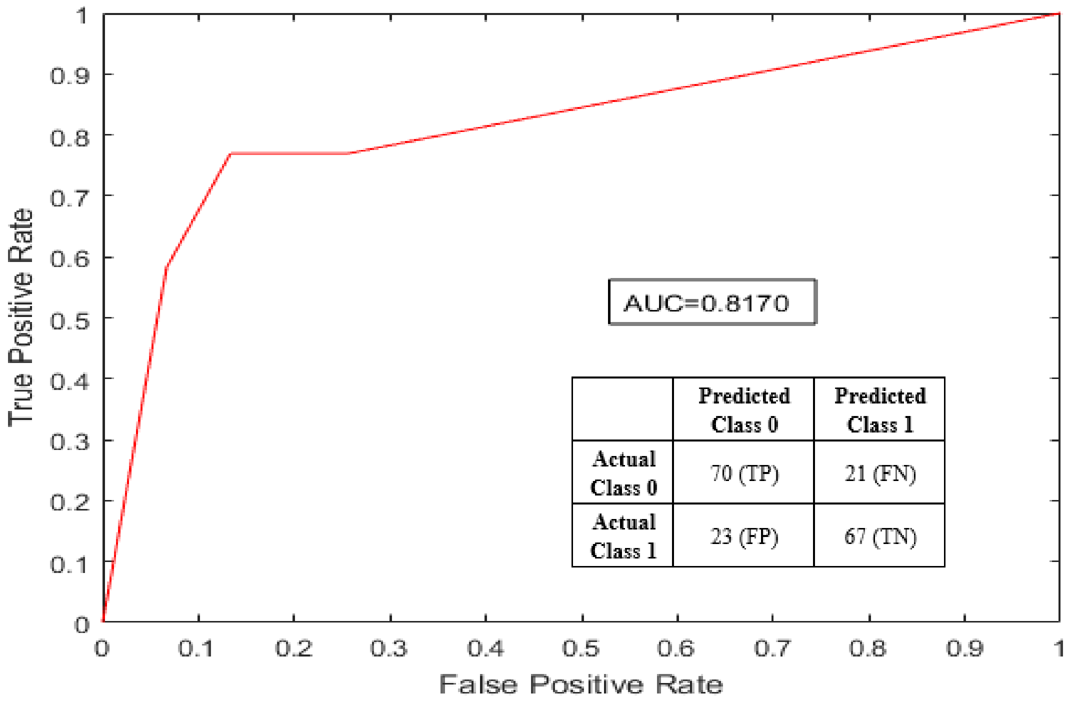
	Predicted Class 0	Predicted Class 1
Actual Class 0	62 (TP)	29 (FN)
Actual Class 1	16 (FP)	74 (TN)

Figure 7.17: ROC curve and confusion matrix of *Algorithm 2* on Linear SVM with image resolution of 0.5.



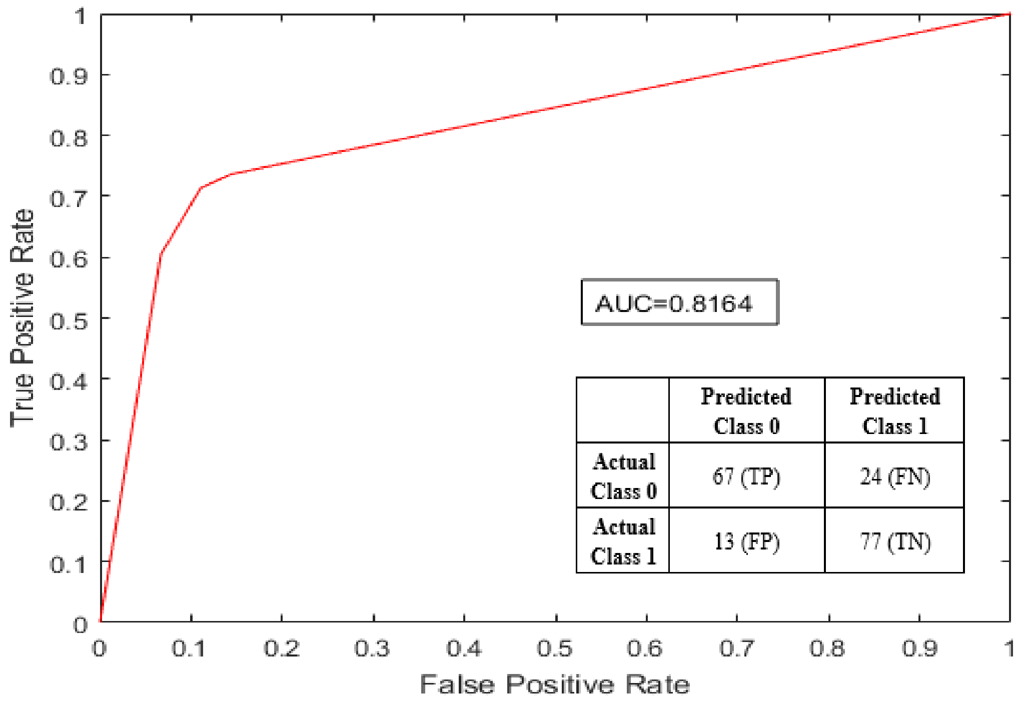
	Predicted Class 0	Predicted Class 1
Actual Class 0	60 (TP)	31 (FN)
Actual Class 1	21 (FP)	69 (TN)

Figure 7.18: ROC curve and confusion matrix of *Algorithm 2* on Linear SVM with image resolution of 0.25.



	Predicted Class 0	Predicted Class 1
Actual Class 0	70 (TP)	21 (FN)
Actual Class 1	23 (FP)	67 (TN)

Figure 7.19: ROC curve and confusion matrix of *Algorithm 2* on Simple Tree with image resolution of 0.5.



	Predicted Class 0	Predicted Class 1
Actual Class 0	67 (TP)	24 (FN)
Actual Class 1	13 (FP)	77 (TN)

Figure 7.20: ROC curve and confusion matrix of *Algorithm 2* on Simple Tree with image resolution of 0.25.

	Linear SVM			**Simple Tree**		
Image Resolution	Sensitivity	Precision	F- Measure	Sensitivity	Precision	F-measure
Resize (0.5)	0.681	0.795	0.734	0.769	0.753	0.761
Resize (0.25)	0.660	0.741	0.698	0.736	0.838	0.784

Table 7.4: Performance measures of *Algorithm 2* when training and testing is done on images of different resolution.

7.3 Answering RQ1

To answer the first research question, initially a null hypothesis and an alternative hypothesis are formulated and the data is analyzed. Based on the results of the test, either the null hypothesis or the alternative hypothesis is accepted. To analyze the performance of both algorithms, Wilcoxon signed rank test is used. A non-parametric test was chosen as we were not sure if the data has a normal distribution and secondly the sample size considered was also small [28]. The authors of Ref.[28] have performed several experiments considering the paired t-test, Wilcoxon signed rank test, sign test and have stated that Wilcoxon signed rank test is better to compare how two classifiers work. Since classifiers are used to classify the images in this study, we have therefore considered using this test. Wilcoxon signed rank test requires the datasets to be dependent. Since the datasets used for both the algorithms are same, even this criterion is satisfied. One more advantage of using Wilcoxon signed rank test is that the effect of outliers is less on this when compared to t-test [28]. In this section, we are using the AUC, and accuracy parameters to test which algorithm performs better. We have considered one tail test as we need to prove that *Algorithm 2* performs better than *Algorithm 1.*

The hypotheses formed based on the research question are as follows

Null Hypothesis: The detection rate of both *Algorithm 1* and *Algorithm 2* is similar.
Alternative Hypothesis: *Algorithm 2* performs better than *Algorithm1.*

The various conducted experiments are concisely summarized below in the form of cases so as to make it easy for further reference.

Case 1: Algorithm using Linear SVM, trained and tested on original images.
Case 2: Algorithm using Linear SVM, trained on original images and tested on images which are reduced by half in resolution (resized by a factor of (0.5)).
Case 3: Algorithm using Linear SVM, trained on original images and tested on images which are reduced by three forth in resolution (resized by a factor of (0.25)).
Case 4: Algorithm using Simple Tree, trained and tested on original images.
Case 5: Algorithm using Simple Tree, trained on original images and tested on images which are reduced by half in resolution (resized by a factor of (0.5)).
Case 6: Algorithm using Simple Tree, trained on original images and tested on images which are reduced by three forth in resolution (resized by a factor of (0.25)).
Case 7: Algorithm using Linear SVM, trained and tested on images which are reduced by half in resolution (resized by a factor of (0.5)).
Case 8: Algorithm using Linear SVM, trained and tested on images which are reduced by three forth in resolution (resized by a factor of (0.25)).
Case 9: Algorithm using Simple Tree, trained and tested on images which are reduced by half in resolution (resized by a factor of (0.5)).
Case 10: Algorithm using Simple Tree, trained and tested on images which are reduced by three forth in resolution (resized by a factor of (0.25)).

Since, we want to prove that Algorithm 2 performs better than Algorithm 1 using one tailed test, "*T*+" is chosen as the test statistic. *T*+ is the sum of the positive signs, *T*- is the sum of the negative signs. The null hypothesis is rejected if the obtained *T*+ value is less than or equal to T_o value (critical value). As n=10, and α=0.05 the value of T_o = 10 from the statistical tables [29].

Case	*Algorithm 1* (a)	*Algorithm 2* (b)	Difference in performance (a-b)	Signed rank
1	0.5807	0.7985	-0.2178	-5
2	0.5862	0.7327	-0.1465	-3
3	0.5862	0.7136	-0.1274	-1
4	0.5718	0.8085	-0.2367	-7
5	0.5430	0.6755	-0.1325	-2
6	0.5569	0.7075	-0.1506	-4
7	0.5659	0.7841	-0.2182	-6
8	0.5306	0.7955	-0.2649	-8
9	0.5248	0.8170	-0.2922	-10
10	0.5281	0.8164	-0.2883	-9

Table 7.5: Performing Wilcoxon signed rank test based on AUC parameter.

When the AUC parameter is considered, the value of *T*+ obtained is 0. As the obtained value of *T*+ is less than 10, the null hypothesis is rejected with 95% confidence level (α = 0.05). So, the alternative hypothesis is accepted which states that *Algorithm 2* performs better than *Algorithm 1.*

Case	*Algorithm 1* (a) (%)	*Algorithm 2* (b) (%)	Difference in performance (a-b) (%)	Signed rank
1	54.69	72.92	-18.23	-5
2	56.9	64.08	-7.18	-2
3	56.9	59.11	-2.21	-1
4	55.8	82.32	-26.52	-9
5	52.48	67.4	-14.92	-3
6	53.03	70.71	-17.68	-4
7	55.8	75.14	-19.34	-6
8	50.83	71.27	-20.44	-7
9	51.93	75.14	-23.21	-8
10	52.49	79.5	-27.01	-10

Table 7.6: Performing Wilcoxon signed rank test based on accuracy parameter.

The value of *T*+ obtained considering the accuracy parameter is 0, which is also less than 10. So, the null hypothesis is rejected even in this case and we can say that *Algorithm 2* performs better than *Algorithm 1* with 95% confidence level (α = 0.05).

RQ1. How can we improve the detection rate of splicing forgery in digital images?

As we can see from the results of both the Wilcoxon signed rank tests, *Algorithm 2* performed better than *Algorithm 1* in both of the cases. So, we can say that we were successful in improving the algorithm. From these results, it can be observed that the YCbCr color transformation and the 2D wavelet transformation are good preprocessing steps on the images. These transformations were not only useful for original images; they also gave similar results on low resolution images. Moreover, and as stated earlier, as the number of extracted features increased for we are considering three channels (YCbCr transformation) instead of one RGB image (in Algorithm 1). This once again proves that extracting good features from the image is always desirable. An algorithm having a stable performance on all sizes of images is a good aspect.

7.4 Answering RQ2

The impact of the image resolution and the classifiers have been examined on both algorithms i.e., *Algorithm 1* and *Algorithm 2.* Experiments were conducted by training and testing the images on the same resolution, and the other set of experiments where the generated model was trained on original images and tested on images of different resolutions (resampling with factors 0.5 and 0.25). These two sets of experiments were compared to know how the performance of the algorithms varied with the change in the training dataset. To compare this aspect, cases 2, 3, 5 and 6 were compared with the cases 7, 8, 9, 10, respectively. Both, the accuracy and the AUC, were used for comparison. From the tables 7.7 and 7.8, it can be seen that performance of *Algorithm 1* improves when we are training with original images and testing on images of different resolutions, but this effect is reversed in the case of *Algorithm 2*.

Algorithm 1		*Algorithm 2*	
Case 2: 56.9%	Case 7: 55.8%	Case 2: 64.08%	Case 7: 75.14%
Case 3: 56.9%	Case 8: 50.83%	Case 3: 59.11%	Case 8: 71.27%
Case 5: 52.48%	Case 9: 51.93%	Case 5: 67.4%	Case 9: 75.14%
Case 6: 53.03%	Case 10: 52.49%	Case 6: 70.71%	Case 10: 79.5%
Mean: 54.8275	**Mean:** 52.7625	**Mean:** 65.325	**Mean:** 75.2625
SD: 2.4036	**SD:** 2.1392	**SD:** 4.9491	**SD:** 3.3629

Table 7.7: Comparing the performance of the algorithms when trained and tested on images of the same resolution v/s when trained on original images and tested on different resolutions based on Accuracy.

Algorithm 1		*Algorithm 2*	
Case 2: 0.5862	Case 7: 0.5659	Case 2: 0.7327	Case 7: 0.7841
Case 3: 0.5862	Case 8: 0.5306	Case 3: 0.7136	Case 8: 0.7955
Case 5: 0.543	Case 9: 0.5248	Case 5: 0.6755	Case 9: 0.817
Case 6: 0.5569	Case 10: 0.5281	Case 6: 0.7075	Case 10: 0.8164
Mean: 0.5681	**Mean:** 0.5374	**Mean:** 0.7073	**Mean:** 0.8033
SD: 0.02169	**SD:** 0.01918	**SD:** 0.02378	**SD:** 0.01622

Table 7.8: Comparing the performance of the algorithms when trained and tested on images of the same resolution v/s when trained on the original images and tested on different resolutions based on AUC.

Cases 1, 2, 3, 7, 8 deal with the Linear SVM classifier whereas cases 4, 5, 6, 9, 10 deal with the Simple Tree classifier. To analyze the performance of the classifiers on the algorithms the mean and SD are calculated. This is shown in tables 7.9 and 7.10. Based on the obtained results, the performance of *Algorithm 1* with Linear SVM is better than that with Simple Tree. But coming to *Algorithm 2* its performance is better with Simple Tree. When the AUC is considered as the parameter only a slight difference in the performance is observed, but when accuracy is considered as the performance measure, the difference could be well observed.

Algorithm 1		*Algorithm 2*	
Linear SVM	**Simple Tree**	**Linear SVM**	**Simple Tree**
0.5807	0.5718	0.7985	0.8085
0.5862	0.543	0.7327	0.6755
0.5862	0.5569	0.7136	0.7075
0.5659	0.5248	0.7841	0.817
0.5306	0.5281	0.7955	0.8164
Mean: 0.56992	**Mean:** 0.54492	**Mean:** 0.76488	**Mean:** 0.76498
SD: 0.023497	**SD:** 0.019732	**SD:** 0.039059	**SD:** 0.068108

Table 7.9: Comparing the performance of the classifiers based on AUC.

Algorithm 1		*Algorithm 2*	
Linear SVM	**Simple Tree**	**Linear SVM**	**Simple Tree**
54.69%	55.8%	72.92%	82.32%
56.9%	52.48%	64.08%	67.4%
56.9%	53.03%	59.11%	70.71%
55.8%	51.93%	75.14%	75.14%
50.83%	52.49%	71.27%	79.5%
Mean: 55.024%	**Mean:** 53.146%	**Mean:** 68.504%	**Mean:** 75.014%
SD: 2.517048 %	**SD:** 1.533763%	**SD:** 6.689681%	**SD:** 6.124033%

Table 7.10: Comparing the performance of the classifiers based on accuracy.

To analyze the performance of algorithms on images of different resolutions and to see how they perform on images of low resolution SD of the accuracy and AUC are computed and analyzed. From the tables 7.7, 7.8, 7.9 and 7.10 it can be seen that the performance of *Algorithm 1* does not have much variation when the resolutions of the images are changed. The variation is the least in the case of *Algorithm 1* using Simple Tree. Coming to *Algorithm 2,* the variation in the performance of the algorithm on different image resolution is less in all cases except in the cases where the training and testing datasets are of different resolutions. When the training and the testing datasets are of different resolutions, the performance of the algorithm is reduced by a huge amount which is affecting the overall standard deviation.

RQ2: What is the impact of image resolution and the classifiers on the performance of the examined image tempering algorithm?

If we consider the impact of the classifiers on the algorithms, the performance of *Algorithm 1* is improved on SVM classifier. Whereas in the case of *Algorithm 2*, it performs better on Simple tree classifier. Coming to the impact of the image resolution, *Algorithm 1*'s performance is enhanced when it is trained on normal images and tested on images of different resolutions. Whereas this is not the case with *Algorithm 2*, its performance is improved when trained and tested on images of same resolution. And it is also observed that there is no much variation on the performance of the algorithms based on the resolution of the algorithms.

7.5 Discussion

The detection rate on test dataset of *Algorithm 2*, on the original images using Simple Tree classifier is 82.32% which is better than some of the existing algorithms, for example the authors of Ref.[7] have stated that their algorithm gives an accuracy of 74%. The performance of the algorithm in Ref.[2], which is stated as 87.75% is better than *Algorithm 2*, but the authors have stated that their algorithm cannot perform well if counter-forensic measures were applied to hide the forgery. The authors of Ref.[2] have also used wavelet decomposition in their algorithm.

The authors of Ref.[7] have stated that their algorithm needs less human interaction. This issue of automation has also been highlighted by the authors of Ref.[16].They have stated that their algorithm is not automated, require human interaction and are trying to work on that. This is not a desirable feature as the person who is operating the process should have knowledge about that algorithm and human operated process are prone to errors. Additionally, for automatic scan of thousands or millions of images (as in the real life) necessitates the need for automation. Taking these aspects into consideration we have made sure that *Algorithm 2* is fully automatic.

Authors of Ref.[15][16] have stated that their algorithm performs well even on images of low resolution and this is a desired feature. When considering *Algorithm 1* and *Algorithm 2*, the variation of performance on images of different resolutions is less, this can be observed through the standard deviation calculated in tables 7.7 and 7.8.

8 CONCLUSION AND FUTURE WORK

8.1 Conclusion

In this thesis we have considered splicing forgery as the area of study. Through this thesis the detection rate of whether an image is forged or clean was improved for the considered algorithm. The impact of the image resolution and the classifiers on *Algorithm 1* and *Algorithm 2* was also analyzed. Implementation and experimentation are the research methods used in this research. Implementation was used to answer RQ1 and experimentation is used to answer RQ2.

Statistical analysis was used to analyze the data obtained through the experiments. Wilcoxon signed rank test was conducted to compare the performance of the algorithm to check if we were successful in improving *Algorithm 1.* Based on the results of the Wilcoxon signed rank test, we were able to tell that *Algorithm 2* performs better than *Algorithm 1* with 95% of confidence.

Through this thesis, we have also analyzed the performance of the algorithms on different resolutions and classifiers. Three types of resolutions were considered, original images, images which were reduced to half the size, and images which were reduced to quarter of the original size. When analyzing the performance of algorithms on low resolution images, it was observed that the performance of *Algorithm 1* was improved when trained on original images and tested on other resolution, whereas the *Algorithm 2*'s performance was better when trained and tested on same image resolution. To analyze how the performance varied over different resolutions the SD was considered, and it was observed that there is no much difference in the performance of each of the algorithms when considering low resolution images. Coming to the performance of the classifiers on the algorithms, *Algorithm 1'*s performance is improved when using the Linear SVM, whereas *Algorithm 2'*s performance is better with Simple Tree.

8.2 Future Work

For the future work, we can validate *Algorithm 2* on datasets of different types and check how it performs on different types of datasets. As sated earlier, we have not considered or studied how the algorithm performs on images which have undergone counter-forensic measures in the form of contrast enhancement. The authors of *Algorithm 1* have considered this aspect and they have also performed the tests on the tampered image which has undergone counter forensic techniques to hide the contrast enhancement. We have not dealt with these aspects as this research only deals with the splicing forgery detection. That is, in this research, the performance of *Algorithm 2* on counter-forensic measures of contrast enhancement has not been tested. So, these areas which were not studied in this research can be explored in future.

Another possibility is to extend the present algorithm so that it can localize the region where the image has been tampered with. To implement such a feature, we first need to extract the features of the image by dividing the image into blocks and if we find a significant difference between the values of one block with respect to the others in that image, then we can say that the image was tampered with in that region.

REFERENCES

[1] C. N. Bharti and P. Tandel, "A Survey of Image Forgery Detection Techniques," *IEEE International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 23-24 March, pp. 877-881, 2016.

[2] A. Kashyap, B. Suresh, M. Agrawal, and H. Gupta, "Detection of Splicing Forgery Using Wavelet Decomposition," *International Conference on Computing, Communicaton and Automation (ICCCA)*, 15-16 May, pp. 843–848, 2015.

[3] Y. J. Zhang, S. H. Li, and S. L. Wang, "Detecting shifted double JPEG compression tampering utilizing both intra-block and inter-block correlations," *Journal of Shanghai Jiaotong University (Science)*, vol. 18, no. 1, pp. 7–16, 2013.

[4] T. Qazi, K. Hayat, S. U. Khan, S. A. Madani, I. A. Khan, J. Kolodziej, H. Li, W. Lin, K. C. Yow, and C.-Z. Xu, "Survey on blind image forgery detection," *Image Processing IET*, vol. 7, no. 7, pp. 660–670, 2013.

[5] A. De Rosa, M. Fontani, M. Massai, A. Piva, and M. Barni, "Second-order statistics analysis to cope with contrast enhancement counter-forensics," *IEEE Signal Processing Letters*, vol. 22, no. 8, pp. 1132–1136, 2015.

[6] Z. Zhang, G. H. Wang, Y. Bian, and Z. Yu, "A novel model for splicing detection," *Proceedings of 2010 IEEE 5th International Conference on Bio-Inspired Computer Theory Application BIC-TA 2010*, pp. 962–965, 2010.

[7] S. N. Youseph and R. R. Cherian, "Pixel and Edge Based Illuminant Color Estimation for Image Forgery Detection," *Procedia Computer Science*, vol. 46, *International Conference on Information and Communication Technologies (ICICT 2014)*, pp. 1635–1642, 2015.

[8] V. Christlein, C. Riess, J. Jordan, C. Riess, and E. Angelopoulou, "An evaluation of popular copy-move forgery detection approaches," *IEEE Transactions on Information Forensics Security*, vol. 7, no. 6, pp. 1841–1854, 2012.

[9] "Image Manipulation Dataset", www5.cs.fau.de, 2016. [Online]. Available: https://www5.cs.fau.de/research/data/image-manipulation/. [Accessed: 16-Dec-2016].

[10] "Columbia Uncompressed Image Splicing Detection Evaluation Dataset", ee.columbia.edu, 2016 [Online]. Available: http://www.ee.columbia.edu/ln/dvmm/downloads/authsplcuncmp/. [Accessed: 9-Sept-2016].

[11] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, *Digital Image Processing using MATLAB*, 1st ed. Upper Saddle River, N.J.: Pearson Prentice Hall, 2004.

[12] P. Flach, *Machine learning*, 1st ed. Cambridge University Press, 2012.

[13] X. Lin, C. Li, and Y. Hu, "Exposing Image Forgery through the Detection of Contrast Enhancenent," *20th IEEE International Conference on Image Processing (ICIP)* pp. 4467–4471, 2013.

[14] P. Ferrara, T. Bianchi, A. De Rosa, and A. Piva, "Image forgery localization via fine-grained analysis of CFA artifacts," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 5, pp. 1566–1577, 2012.

[15] H. Yao, S. Wang, Y. Zhao, and X. Zhang, "Detecting Image Forgery Using Perspective Constraints," *IEEE Signal Processing letters*, vol. 19, no. 3, pp. 123–126, March, 2012.

[16] M. Iuliani, G. Fabbri, and A. Piva, "Image Splicing Detection based on General Perspective Constraints," *IEEE International Workshop on Information Forensics (WIFS)* pp. 1–6, 2015.

[17] B. Mahdian and S. Saic, "Using noise inconsistencies for blind image forensics," *Image and Vision Computing*, vol. 27, no. 10, pp. 1497–1503, 2009.

[18] X. Pan, X. Zhang, and S. Lyu "Exposing Image Forgery with Blind Noise Estimation," pp. 15–20, 2011.

[19] Tu K. Huynh, Khoa V. Huynh, Thuong Le-Tien, and Sy C. Nguyen, "A survey on image forgery detection techniques," *Proc. 2016 IEEE RIVF International Conference*

on Computing and Communication Technologies - Research, Innovation, and Vision for the Future (RIVF), 25-28 Jan, pp. 71-76, 2015.

[20] "Statistics and Machine Learning Toolbox - MATLAB & Simulink", Se.mathworks.com, 2016 [Online]. Available: http://se.mathworks.com/products/statistics/?requestedDomain=www.mathworks.com. [Accessed: 6-Dec-2016].

[21] "Image Analysis Processing & Protection Group." [Online]. Available: https://iapp.dinfo.unifi.it/index.php/english/materials_en/source-code_en. [Accessed 6-sept-2016]

[22] "Single-level discrete 2-D wavelet transform - MATLAB dwt2 - MathWorks Nordic." se.mathworks.com, 2016 [Online]. Available: http://se.mathworks.com/help/wavelet/ref/dwt2.html. [Accessed 6-Dec-2016]

[23] A. Ford and A. Roberts, *Colour Space Conversions*, Westminster University, London, 1998.

[24] M. Berndtsson, J. Hansson, B. Olsson, and B. Lundell, *Thesis Guide - A Guide for Students in Computer Science and Information Systems*. 2008.

[25] Wohlin, Claes, *Experimentation in software engineering*. 1st ed. Berlin: Springer. 2012.

[26] D T Campbell, and J. C. Stanley, *Experimental and Quasi-Experimental Designs for Research*. Vol. 4. Rand McNally, 1971.

[27] R. Feldt and A. Magazinius, "Validity Threats in Empirical Software Engineering Research - An Initial Survey," *Proceedings of the Software Engineering and Knowledge Engineering Conference*, August, pp. 374–379, 2010.

[28] J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *Journal of Machine Learning Resesearch.*, vol. 7, pp. 1–30, 2006.

[29] F. Sani and J. Todman, "Appendix 1: Statistical Tables," *Experimental Design Statistics for Psychology: A First Course*, vol. 2, no. 6, pp. 183–196, 2008.