Spring 6-8-2016

Image Spam Analysis

Annapurna Sowmya Annadatha
San Jose State University

Image Spam Analysis

A Project

Presented to

The Faculty of the Department of Computer Science

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Annapurna Sowmya Annadatha

May 2016

The Designated Project Committee Approves the Project Titled

Image Spam Analysis

by

Annapurna Sowmya Annadatha

APPROVED FOR THE DEPARTMENTS OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

May 2016

ABSTRACT

Image Spam Analysis

by Annapurna Sowmya Annadatha

Image spam is unsolicited bulk email, where the message is embedded in an image. This technique is used to evade text-based spam filters. In this research, we analyze and compare two novel approaches for detecting spam images. Our first approach focuses on the extraction of a broad set of image features and selection of an optimal subset using a Support Vector Machine (SVM). Our second approach is based on Principal Component Analysis (PCA), where we determine eigenvectors for a set of spam images and compute scores by projecting images onto the resulting eigenspace. Both approaches provide high accuracy with low computational complexity. Further, we develop a new spam image dataset that should prove valuable for improving image spam detection capabilities.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. Mark Stamp, for his encouragement, patience, and continuous guidance throughout my project and graduate studies as well.

I am very grateful to the committee member, Dr. Thomas Austin for being extremely helpful and for his valuable time.

My thanks also go to the committee member, Fabio Di Troia, for monitoring the progress of the project and suggestions.

Last, but not least, I would like to thank my husband, Prashanth, for his support and encouragement which made this possible. My parents, Usha Rani and Amara Prabhu, receive my deepest gratitude and love for all the years of support that provided the foundation for this work. And I also thank my son, Rohan, for his understanding and love in his little own way during the past few years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Electronic mail, commonly known as e-mail, is the most widely used application for exchanging digital messages across the globe [17]. The usefulness of e-mail is threatened by spamming, which is defined as unsolicited bulk e-mail. Spam e-mail may, for example, contain advertisements, links to phishing websites or malware attached as executable files. Spam is illicit in nature, targeting numerous recipients around the world.

According to Symantec [36], during 2015 spam accounted for approximately 60 percent of all inbound e-mail, making spam filtering necessary for the continued utility of e-mail. The first generation of spam e-mail contained only text messages. As spam filters became more effective, spammers developed various techniques to bypass such filters. Image spam, which consists of spam text embedded within an image, is used by spammers as a means of evading detection. It has been reported that image spam is around 40 percent of the overall spam traffic, and this percentage continues to rise. Figure 1 shows image spam as a percentage of overall spam volume for the year 2015 as reported by Trustwave [34].

Previous approaches to detecting image spam consist of extracting various image properties and classifying the images as either spam or ham (i.e., non-spam), using machine learning techniques. This previous work either obtains low accuracy [9, 12, 39] or has an unrealistically high computational complexity [7, 19].

In this research, we develop and analyze two techniques for distinguishing spam images from ham. Specifically, we extract high-level and low-level image properties

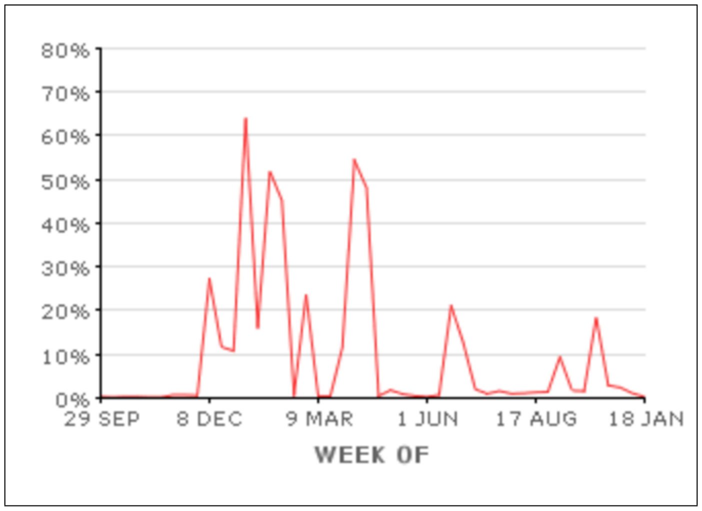


Figure 1: Image spam as a percentage of overall spam for the year 2015

and use a feature selection algorithm based on Support Vector Machines (SVM) to minimize the computational complexity of the model. We also consider a technique based on Principal Component Analysis (PCA), and compare these two approaches. These techniques both perform very well against current spam images. We then use insights gleaned from our detection techniques to develop a more challenging class of spam images. This dataset should prove valuable for improving image spam detection capabilities.

The remainder of this paper is organized as follows. Chapter 2 provides an overview of image spam and related work on detection techniques. Chapter 3 describes Support Vector Machines and the presents our selective approach based on this technique. Chapter 4 briefly explains the face-recognition technique called Eigenfaces, followed by our method of Eigenspam. Chapter 5 gives the implementation details for our techniques. Experimental results and performance analysis are presented in Chapter 6. Finally, Chapter 7 provides the conclusion and possible future enhancements to the current study.

CHAPTER 2

Background

Spamming refers to sending unsolicited electronic messages to a large group of users arbitrarily. The evolution of the spam began with the idea of advertising various products. Later, it had been practiced as a prank in the gaming field and also advanced to include some malicious activities. But, the focus of spamming has particularly moved to e-mail, where it prevails today. Since the advertisers have no operating costs except to obtain the servers, mailing lists and the infrastructures, it has proven to be economically viable. And as there is no particular barrier, spammers have emerged numerously and are not being held for their mass mailings. In the year 2015, the predicted figure for spam messages is around seven trillion [18]. Public and several organizations had to borne the costs that include loss of productivity and bandwidth.

2.1 Types of Spam

Spam takes different forms based on the targeted media such as email spam, mobile spam, social networking spam, gaming spam [20].

Email spam is the unsolicited bulk email and the most commonly encountered spam. Identical messages are sent to numerous recipients by email; that may contain advertisements, links to phishing sites or some malware attached. Different email spam techniques include blank spam, image spam, and text spam. Initially, spam was only text based, i.e., spam email contained text in the body of the message. As the filters could classify these text messages, spammers introduced a sophisticated image-based technique. Image spam is an obfuscation method in which the text is embedded in an image and displayed in the mail. Blank spam as the name suggests does not contain any message. It is a kind of dictionary harvest attack to collect valid email addresses.

Social spam targets the most common social networking sites such as Facebook and Twitter. Spammers hack into the accounts and send some false links to the user's trusted contacts.

Mobile spam aims at the text messaging service of a mobile device. "SpaSMS" is coined to describe the spam SMS [33]. Mobile spamming not only causes inconvenience to the customers but also could cost the incoming message. Though a huge number of people use the mobile phone, there has been a considerably less amount of mobile spam because of the cost involved with sending the messages.

Gaming Spam is a form of message flooding to the players via chat rooms or public discussion areas. It is particularly common for the spammers to sell game related items for real-world money or in-game currency.

2.2 Image Spam

Image spam is a class of email spam, which has evolved as an obfuscation technique to bypass the traditional text-based spam filters. The text is embedded into a graphical image which appears as the main body of the message. The method to detect text spam was to identify keywords from the text and block the message. Thus, more advanced form of email spam was developed by the spammers. The process to determine if an image in an email is spam or ham is as follows [16]:

- First, if the sender of the email is in the black list, mark it as spam. If the email is marked as spam, then determine if the message contains an image. Though the email is identified as spam, we can not conclude that the image included is spam. Thus, a spam filter is required to classify it as spam or ham.

- If an email is marked as legitimate using white-list, the image contained in the message is filtered. If it is found to be spam image, the image is displayed only with users' consent.

- If a new email address is encountered and it contains an image, the image has to be filtered. If the image is detected to be spam, the image is not displayed. A choice is given to the user to mark it as spam.

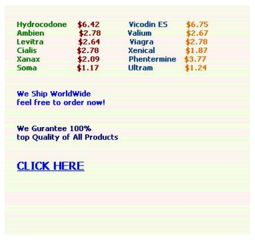
2.2.1 Types of Image Spam

Image spam is heavy in content and has progressed in different forms to constantly challenge the conventional filters: Pure text image, Randomized image, Mixed image.

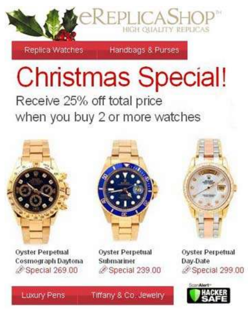
Pure text image is the first generation spam image and rich in text. It contains a pure text file embedded as a picture in the message as shown in Figure 2a. Since most e-mail clients rendered graphical images automatically, it could deliver the intended message successfully. A filtering technique that uses Optical character recognition (OCR) has been employed to identify such spam. Words from the image are extracted using OCR, and the text is passed to the traditional text-based filter.

Randomized image is introduced to increase the difficulty of filtering spam using OCR. Spammers created images by adding noise, a thin background of random artifacts, or by rotating the image slightly. These changes do not influence the readability of the users but significantly affect the output of the OCR based filter. The spammers also carried out some unnoticeable changes such as shading the border, adding tiny streaks with almost random patterns to bypass the filters that are based on Hash signatures. An example of this kind is shown in Figure 2d.

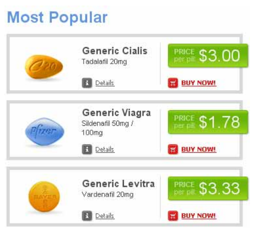
A Mixed image is a form of spam that is split into multiple parts to contain both text and some related picture or an animation. These forms appear to be very close to genuine and are challenging to be detected by most of the filters. Images shown in Figure 2b and Figure 2c.



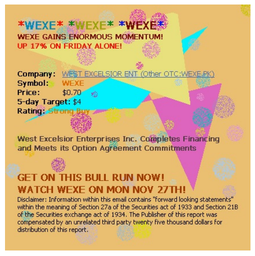
(a)



(b)



(c)


WEXE *WEXE* *WEXE*
WEXE GAINS ENORMOUS MOMENTUM!
UP 17% ON FRIDAY ALONE!
Company: WEST EXCELSIOR ENT (Other OTC:WEXE.PK)
Symbol: WEXE
Price: $0.70
5-day Target: $4
Rating: Strong Buy
West Excelsior Enterprises Inc. Completes Financing and Meets its Option Agreement Commitments
GET ON THIS BULL RUN NOW!
WATCH WEXE ON MON NOV 27TH!
Disclaimer: Information within this email contains "forward looking statements" within the meaning of Section 27a of the Securities act of 1933 and Section 21B of the Securities exchange act of 1934. The Publisher of this report was compensated by an unrelated third party twenty five thousand dollars for distribution of this report.

(d)

Figure 2: Spam Images

2.2.2 Content Categories

Image spam can contain the following categories [20]:

- Advertisements — The content includes some details to promote a product or some online shopping offers for various
- Pornography — The image contains obscene pictures or products.
- Phishing links — Images contain a link to websites that would attempt to acquire their usernames and passwords or can also contain a link to malicious websites.
- Scams — Typically promises a set of victims with some significant share of a

substantial amount of currency for a small payment. This fee helps the fraudster to collect a huge amount and later disappear or creates a series of the victims.

2.3 Image Spam Detection

Image spam detection techniques fall into two categories: Content-based and Noncontent-based.

2.3.1 Content-based Filter

Content-based filters inspect the image content for specific keywords typically used in the spam text. And then may use pattern recognition technique to track particular behavior or pattern. One of the early content based filters is Spam Assasin [32], which uses OCR to extract text from the images and analyze the text using traditional text-based filters.

2.3.2 Noncontent-based Filter

The noncontent-based filter uses different properties of the image to classify the spam image. The earlier work included extracting simple metadata features and color properties of an image. This kind of filter relies on the fact that the genuine image possesses distinct characteristics compared to a computer generated image with text.

2.4 Related Work

Image spam detection has stirred interests of many researchers since the time of its birth. Various detection techniques and algorithms have been explored for an efficient classification. In this section, we present some works in the scientific literature that dealt with the image spam detection.

One of the classical techniques for the content-based filters was Spam Assasin [32], which uses Optical Character Recognition (OCR) to recover the text from the spam images. OCR is a mechanism for electronic conversion of images with any text (handwritten or printed) into machine-encoded text. The text-based filters are used for the further analysis of the extracted text. To bypass these filters, spammer used various obfuscation techniques like making the text blur which practically effected the quality of the existing filters.

The limitations of the OCR-based detection have been defeated by non-content based email spam filtering. One of the early works in this area is Learning fast classifiers for image spam [7]. Authors in this paper propose a fast and robust detection method by extracting a set of features based on simple file properties and metadata. They employed Maximum Entropy, a discriminative model to evaluate the feature set. They included two other classifiers such as Naive Bayes and ID3 decision trees for comparison and also to represent a different learning approach. The results showed that Maximum Entropy outperformed Naive Bayes while decision tree is a close second. This research also offers Just-in-Time (JIT) extraction which focuses on extracting features based on each image. The basis of the feature selection is a greedy inclusion by mutual information technique. A JIT decision tree is constructed to evaluate the feature sets. The overall system demonstrated an efficient classification.

Image spam hunter [12] proposed a detection method based on probabilistic boosting tree considering global image features like color and gradient histogram. They suggested a learning-based prototype system to differentiate spam from normal images. The system works by first clustering the spam into groups based on a similarity measurement of color and gradient oriented histograms. Then build a probabilistic boosting tree on the training set (chosen from the clustered groups) to distinguish the images. Results obtained from SVM classifier are considered as a baseline for the comparison of their system. The proposed technique achieved an accuracy rate of 89.44% with 0.86% false positive rate.

A comprehensive server to client side approach for image spam detection [9] applied the clustering technique at the server end to segregate genuine and spam. Further, feature-based active learning methods have been used for classification at the client side. The server-side approach involves a sparse nonnegative representation of a similarity measure. This similarity measure is combined with a spectral clustering algorithm for analysis of spam images. SVM active classification technique is applied at the client-side to filter the spam images that have survived during the server-side detection. This system extracted a large feature set and thus increased the computational complexity though have ensured an accuracy rate of 99%.

Detecting image spam based on file properties, Hough transform, and histogram [39] focussed the detection constrained by specific image attributes. The authors of this paper extracted features related to three different domains and compared their individual performance. The first approach considers features related to file properties like file format, width, height, and aspect ratio that are derived at a low computational cost. The second method uses classification based on color histograms of the image. The final approach uses Hough transforms for the classification. Hough transform is a method to find different shapes present in the images using edge detection. According to their experimental results, an approach using file properties eliminates around 80% of image spam and the method using the histogram to implement distance measure reduces 84% of the spam images whereas the Hough transform method achieves 88% of accuracy and also minimizes the false positive rate

Fuzzy Inference System based Image Spam Detection Technique (F-ISDS) [19] introduced a server side solution to extract multiple domain features and further classify based on fuzzy inference system. F-ISDS employs dimensionality reduction based on Principal Component Analysis (PCA) to reduce the number of features by mapping them to their principal components. Further, Linear Regression Analysis is used to model the relationship between the principal components and the extracted features. Based on the model, membership functions are designed for a Fuzzy Inference System classifier. Though this work focused on an extensive range of features, obtained an accuracy rate of 85%.

This comprehensive study of the existing research motivated us to develop an efficient system for the image spam detection using various machine learning algorithms.

CHAPTER 3

SVM-based Detection

A large number of classification algorithms have been applied to spam detection area earlier. Among them, Support Vector Machine (SVM) [15] is a useful technique and famous for its good generalization performance. Previously, SVM has been widely used for text-based spam detection. As SVM provides a useful analysis by separating the hyperplane, in this work, we employed SVM for classification. In the next sections, we present an overview of SVM technique followed by our approach for image spam detection based on SVM.

3.1 SVM Overview

Support vector machines is a class of algorithms based on optimization and is used for binary classification. SVM is a supervised learning technique, which means that it requires labeled training data. That is, we must use data that has been categorized and labeled in advance. The algorithm uses the labeled data to train a model and find an optimal hyperplane that separates the two classes of the data by acting as a threshold. This technique is based on the following intuitive ideas [35]:

- **Maximize the margin**: Consider a labeled training data, we attempt to find a hyperplane that separates the data. Since it is binary classification, we try to divide the two classes with a maximum margin of separation.The margin is determined by the minimum distance between the hyperplane and any point in the training set. For example, the hyperplane is denoted by the yellow line in Figure 3. The black arrows indicate the nearest elements called the support vectors.

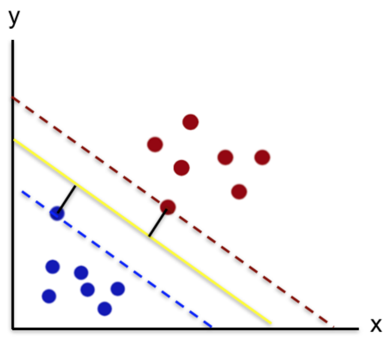


Figure 3: Maximum margin of seperation

- **Work in higher dimension**: For a given data, if a hyperplane exists, we recognize the problem as linearly separable. To deal with the cases where data is not linearly separable, SVM employs a technique to move the input space to a higher plane giving additional space to find the hyperplane. For example, the data on the left side of the Figure 4 cannot obtain a linear separation. Though a parabola is possible, we can also use a transformation technique (ϕ) to map input space to feature space and determine the hyperplane.

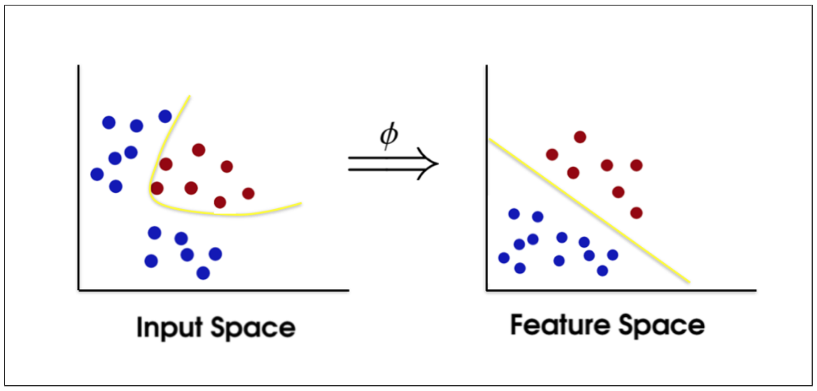


Figure 4: Linear seperation

- **Kernel trick**: This is essentially a mapping function that transforms the data into higher dimensions. Kernel trick makes it easy to compute a hyperplane in the feature space. Figure 5 shows an example where the input space on the left is not linearly separable, but after applying the transformation based on kernel trick, we can easily construct a hyperplane.

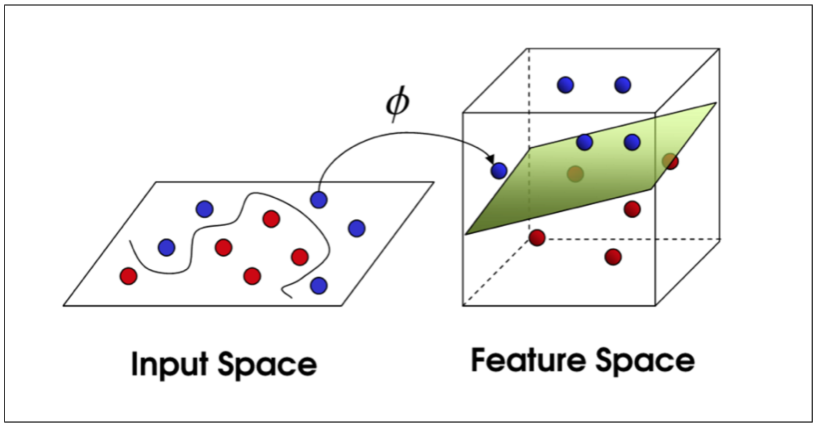


Figure 5: Tranformation to higher dimension

3.1.1 SVM Algorithm

SVM algorithm functions in two phases: Training Phase and Testing Phase. In the Training Phase, we create a model by training on a labeled dataset. In the Testing Phase, we apply the generated model to the test dataset. The test phase determines the accuracy of the given classifier. The SVM training and testing algorithms can be generalized as follows [35].

3.1.1.1 Training Phase

The SVM training phase determines the equation of a separating hyperplane that maximizes the margin. The obtained hyperplane will separate the feature space into two sets. As a result, we can classify any points during the testing phase. The generic process involves solving Lagrangian Duality [35]. The simplified steps of SVM Training Phase algorithm is given below.

1. Consider a set of labeled training data consisting of points $X_1, X_2, \ldots, X_n$ and a corresponding set of classification $z_1, z_2, \ldots, z_n$, where $z_i \in \{-1, 1\}$.

2. Select a kernel function K and a parameter C, which specifies the permissible number of classification errors.

3. Solve the optimization problem

$$\begin{aligned} \text{Maximize: } L(\lambda) &= \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j z_i z_j K(X_i, X_j) \\ \text{Subject to: } \sum_{i=1}^{n} \lambda_i z_i &= 0 \text{ and } C \geq \lambda_i \geq 0 \text{ for } i = 1, 2, \ldots, n \end{aligned} \tag{1}$$

   to obtain λ_i and b.

3.1.1.2 Testing Phase

Given a trained SVM, we score a data point by determining on which side of the separating hyperplane it resides. The process involves the following steps.

1. Given a datapoint X, we compute score

$$f(x) = \sum_{i=1}^{s} \lambda_i z_i K(X_i, X) + b \tag{2}$$

   where s is the number of support vectors and is much smaller than n. This can also be written as

$$f(x) = w \cdot x + b \text{ where } w = \lambda_i z_i K(X_i, X). \tag{3}$$

   This w is called the weight vector which represents the support vectors orthogonal to the hyperplane in the Figure 6.

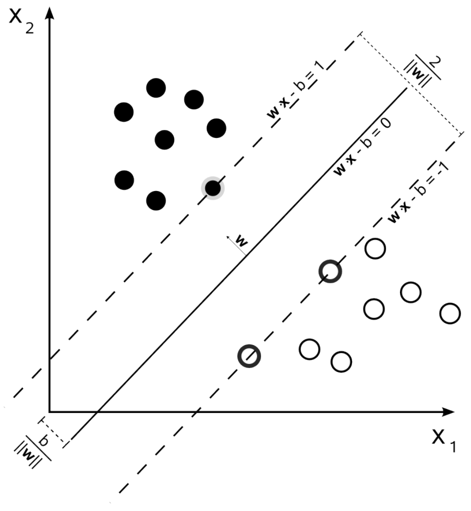


Figure 6: SVM weight vector

2. Classify X according to

$$c(X) = \begin{cases} 1 & \text{if } f(X) > 0 \\ -1 & \text{otherwise.} \end{cases} \tag{4}$$

3.1.2 Kernel Functions

A Kernel function K, accepts inputs as X_i and X_j and produces an output that is the inner product of their respective images, $\phi(X_i)$ and $\phi(X_j)$ [5]. The function is given by

$$K(X_i, X_j) = \phi(X_i) \cdot \phi(X_j)$$

which represents a dot product of the input data points mapped onto the higher dimensional feature space by ϕ transformation. There are different Kernel functions used in practice. SVM classifier allows the user to specify one of the following kernels:

- **Linear Kernel:**

$$K(X_i, X_j) = X_i \cdot X_j$$

- **Polynomial Kernel:**

$$K(X_i, X_j) = (X_i \cdot X_j \ + \ 1)^p$$

- **Radial basis function:**

$$K(X_i, X_j) = e^{-(X_i - X_j)(X_i - X_j)/(2\sigma^2)}$$

- **Two Layer perceptron:**

$$K(X_i, X_j) = \tanh(\beta_0(X_i \cdot X_j) + \beta_1)$$

  Choosing the correct kernel is a nontrivial task. The classification generally depends on tuning the kernel parameters no matter which kernel is chosen.

3.2 Selective Approach using SVM

In this paper, we present a novel approach based on the SVM algorithm that allows a sensitive feature selection apart from the traditional feature extraction. Then, creates a model for fast and accurate classification. The baseline of the system is the fact that the characteristics of a spam image, which is computer generated are distinct from a genuine camera generated image [7]. Our empirical evaluation shows that our model yields high accuracy with a reduced computational cost. With this approach, we extract a broad set of features from various domains. A feature selection algorithm based on SVM weights has been used to select a subset containing highly distinguishing features to make the classification easy. Thus, our model employs Feature Extraction, Feature Selection, and Classification, which we discuss further in detail.

3.2.1 Feature Extraction

In this feature extraction module, we detect and isolate various desired portions or shapes of a digitized image as a compact feature descriptor or feature vector [26]. We extract an effective set of 21 discriminant image features. These features have been identified from various image property domains that classify a computer-generated image and a camera-generated image. Various features have been employed in the previous related work for noncontent-based filtering. After rigorous simulation and empirical analysis, we chose the following image statistics from the literature to integrate color, shape, metadata, noise and appearance properties.

- **Color Domain:** Considering that the spam images are composed of fewer color shades compared to the genuine images, we extract color related properties to classify the images. We compute the entropy of the color histogram as the first image feature. By quantizing each color band in the RGB space, the color histogram depicts the distribution of colors in an image. We further set up a histogram for the three color channels separately and then compute the mean, variance, skewness and kurtosis for each, which adds another 12 feature statistics [9].

- **Texture Domain:** Local binary pattern (LBP) [22] labels the pixel of an image by thresholding the neighborhood pixels. This shows the similarity or differences between the neighboring pixels. The entropy of the LBP histogram is calculated as one texture feature [9]. The natural images are likely to have more information in this feature vector as they are captured with a wide variety of backgrounds and foregrounds.

- **Shape Domain:** The histogram of oriented gradients (HOG) descriptor is the local object appearance and shape within an image. It can be described by the distribution of the intensity gradients; that is the directional change in the intensity of the image. The entropy of the gradient magnitude orientation histogram is the first shape feature [11]. Then we use edges of the image to compute the next two image features. The curved line segments formed by a set of points where image brightness change sharply are called edges [3]. The total number of edges and the average length of the edges are examined for our classification [10]. As spam images are generally a combination of text and images, they are supposed to have more number of edges compared to the ham images.

- **Metadata Domain:** Compression ratio and Aspect ratio of the images are calculated as the two metadata features [39]. Compression ratio captures the amount of compression achieved by calculating the ratio of pixels in an image to an actual image size. Aspect ratio is a proportional ratio between width and height of an image. Spammers try to compress the picture to minimize the size of the email which discriminates from a genuine one.

- **Noise Domain:** Image noise is the random change of brightness or color information in images and is an aspect of electronic noise. The effects of the noise content are analyzed by computing entropy of noise and signal to noise ratio which is the measure of sensitivity [19]. Genuine images are captured in varied backgrounds tend to have more noise than spam images unless noise is artificially added.

Table 1 gives the list of all the features with the abbreviations as used in the rest of the paper.

Table 1: List of Image Features

Feature No.	Feature Name	Abbreviation used
f1	Entropy of Local Binary Pattern Histogram	LBP
f2	Entropy of Histogram of Oriented Gradients	HOG
f3	Number of Edges	Edge no
f4	Average length of Edges	Edge len
f5	Compression Ratio	Comp
f6	Aspect Ratio	Aspect
f7	Signal to Noise Ratio	SNR
f8	Entropy of Noise	Noise
f9	Entropy of Color Histogram	Color hist
f10	Mean of Red Channel	Mean1
f11	Mean of Blue Channel	Mean2
f12	Mean of Green Channel	Mean3
f13	Variance of Red Channel	Var1
f14	Variance of Blue Channel	Var2
f15	Variance of Green Channel	Var3
f16	Skewness of Red Channel	Skew1
f17	Skewness of Blue Channel	Skew2
f18	Skewness of Green Channel	Skew3
f19	Kurtosis of Red Channel	Kurt1
f20	Kurtosis of Blue Channel	Kurt2
f21	Kurtosis of Green Channel	Kurt3

3.2.2 Feature Selection

In the feature selection module, we select a subset of highly discriminant features for our model construction [2]. The aim is to mute out features that are not useful to the existing features during classification. The feature selection could invariably reduce the computational complexity and increase the speed. There are various algorithms for ranking the features. In this paper, we explored a different perspective of Support Vector Machines(SVM) for feature ranking. Feature selection using SVM was earlier used for identification of gene subsets in biological research [14]. In this section, we describe two feature selection strategies that we analyzed.

3.2.2.1 Feature Ranking using SVM Weights

Recall that in linear SVM, w is the weight vector that represents the coordinates of the support vectors orthogonal to the resultant hyperplane in equation (3). The dot product of this weight vector with any point from the test set gives the direction of the predicted class. So using the weight vector we find the most useful feature for separating the data with the fact that the hyperplane would be orthogonal to that axis. The absolute size of this weight coefficient relative to the other points gives an indication of how useful the feature is during the classification [24]. Our first approach for feature selection is based on the weights as ranking criteria to limit the number of features. Weight vectors are evaluated by training the model using linear SVM. Then n features with the largest absolute value of the weight are marked as the most important features. This is intended to reduce the computational effort during the test phase. The steps for this algorithm to select a subset of n features are as follows:

1. Train the classifier using Linear SVM using the feature set
2. Compute the SVM weights for all the features
3. Select the first n features with the largest weights
4. Create a model based on the subset

3.2.2.2 Recursive Feature Elimination using SVM Weights

The usual feature ranking criteria becomes very sub-optimal when it comes to removing several features at a time, which is necessary to obtain a small and accurate feature subset. This problem is overcome by using an iterative procedure called Recursive Feature Elimination (RFE) [14]. RFE is designed to recursively remove the attributes and build a model on the remaining attributes. It relies on the model accuracy to identify the combination of attributes that contribute the most to predicting the target attribute. Here we used linear SVM as the model. The steps for the RFE algorithm are as follows:

1. Train the classifier using Linear SVM using the feature set
2. Compute the ranking criterion using SVM weights
3. Remove the feature with smallest ranking criterion
4. Repeat steps 1 to 4 until number of features in the subset matches the criteria

3.2.3 Classification

The model generated after feature selection is applied to the test set. Selected features are extracted from the test set images. Then the SVM technique is used for classifying the two classes of spam and ham images. Different kernel functions have been applied during the empirical testing, and the analysis is presented in Chapter 6.

3.3 New Spam Data

In previous section, we discussed the SVM weight vectors which reveal the importance of the features by training a model. The weight vectors not only help to minimize the computational effort but also expose the determinant characteristics of the spam images. This fact motivated us to create a new set of spam images that would attempt to conceal all the visible traits of the original spam images. This has been explained in detail in Chapter 6.

CHAPTER 4

PCA-based Detection

In this chapter, we focus on PCA-based image spam detection strategy. In comparison to the earlier detection technique, this approach also depends on the structural entropy score but does not involve feature extraction for the images. Principal Component Analysis (PCA) is a linear transformation method that finds the directions (principal components) of the data and maximizes the variance. PCA relies on eigenvector analysis which is also achieved using Singular Value Decomposition (SVD) [35].

First, we briefly discuss the mathematical model of eigenvalues and eigenvectors that helps us for the theoretical discussion of PCA in the following section. Then we review Eigenfaces, an application of PCA to the well-known facial recognition problem from [37], which forms the basis for our "Eigenspam" detection technique. We present the implementation and experimental analysis in the coming chapters.

4.1 Eigenvalues and Eigenvectors

Eigenvalues and eigenvectors are prominently used in the interpretation of linear transformations. The term *eigen* means "unique to", or "peculiar to" in the sense of "characteristic" [41]. It was initially designed to analyze principal axes of the rotational motion of rigid bodies.

In linear algebra, eigenvector is a non-zero vector x for a square matrix A that satisfies $A = \lambda x$ where λ is scalar called eigenvalue. Eigenvector x of a matrix A implies that A stretches the eigenvector x by λ, without changing the direction. Figure 7 depicts a vector x and its geometrical transformation λx. It shows that for a particular matrix A and a vector x, the effect of Ax is same as scalar multiplication of eigenvector x with a magnitude given by its eigenvalue λ. Considering each eigenvalue is associated with an eigenvector, the relative importance of an eigenvector is given by the magnitude of its corresponding eigenvalue and hence can be used for dimensionality reduction of data by removing the unimportant directions.

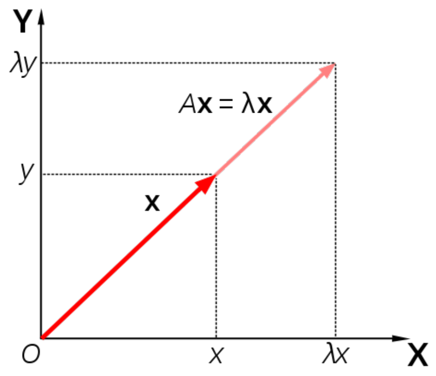
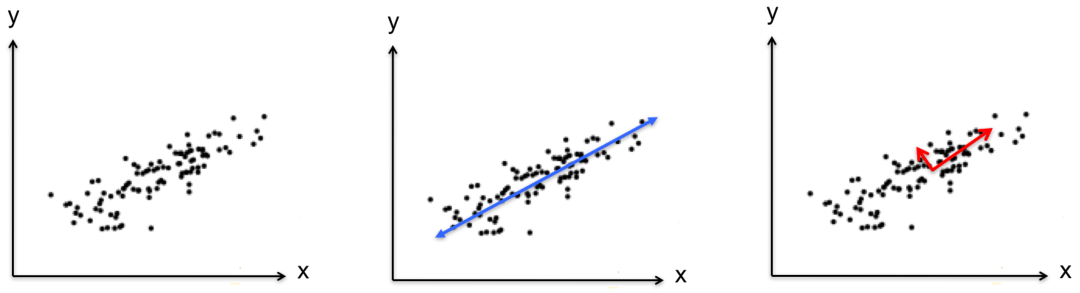


Figure 7: Eigenvector [6]

4.2 Principal Component Analysis

PCA is a linear algebra technique which helps us find the most significant directions in the data that enables to reduce the dimensionality of the problem. In simple terms, PCA is a way to identify patterns in the data and express them to identify the similarities and differences. If the data is of higher dimension, it is hard to find the patterns without any graphical representation. PCA minimizes this problem by reducing the dimensionality. The main idea of PCA for a given dataset S with n variables is to define this dataset with a smaller set of elements that are linear combinations of the original values. These elements are called the principal components.

For example, consider the a scatterplot of the scores from an experiment as shown in Figure 8a. The minimal spanning set of the data is not very informative, so we try to reduce the dimensionality. We can use a regression line as shown in Figure 8b, which is the best-fitting straight line to the set of points. This regression line reduces the dimensionality by obtaining the significant information but there is also some loss of information. PCA also uses linear regression but provides a better basis that reveals the structure for viewing the data. Figure 8c illustrates this basis, where the direction determines the structure and the length measure gives the variance of the data in the given direction. PCA thereby eliminates the less informative directions proportional to the variance to reduce the dimensionality.



(a) Experiment Data (b) Regression Line (c) PCA Basis

Figure 8: PCA Example

The process of dimensionality reduction using PCA consists of following steps [31].

1. Consider an experimental data as matrix $A_{m \times n}$, where n is the number of experiments, and m is the number of measurements per experiment.

2. Compute the mean of each row and subtract it from the each element of the corresponding row.

3. Construct a covariance matrix $C = \frac{1}{n}AA^T$. Each element on the principal diagonal of this matrix is just the variance of each element of the vector while

the off-diagonal element is the covariance. We use the covariance matrix [40] to determine the principal components.

4. To diagonalize the covariance matrix, compute the eigenvalues and eigenvectors of the matrix C.

The diagonalized matrix reveals the inherent structure of the data. Also, the dominant eigenvalues correspond to the most informative directions of the data. Thus, we use this information to reduce the dimensionality of the problem. In the next section, we discuss a classical application of PCA for facial recognition.

4.3 Eigenfaces

Eigenfaces [37] is a technique that is widely used for face recognition and is based on PCA. Eigenfaces approach interprets the complex problem of face recognition as a 2-dimensional problem by assuming faces to be upright and ignoring the geometric features of the face, which makes it computationally simple.

4.3.1 Face Recognition

For this approach, first, a set of training images is considered with the faces of images centered. A covariance matrix is constructed with the pixel values from the images. Eigenvectors of this covariance matrix are computed to measure the similarity between different images. These vectors are referred to as “eigenfaces”. The eigenfaces exhibit significant features of a face but do not certainly present intuitive features such as lips, eyes, etc. The feature space defined by these eigenfaces is termed as “face space”.

The images from the training set are projected onto the face space, and a weight vector for each image in the training set is obtained. These set of weights are used to score any image. Consider an unknown image is given for classification, the image is first projected onto the face space. The Euclidean distance to each of the training images from the weight vector of the projected image is computed. The minimum of these distances defines the score. If the score is small, then the image is considered to be the closest match to a class in the training set, else the image does not belong to that class.

To illustrate this process, consider the images in Figure 9 to be the training set. The training images have been downloaded from Yale face database [42] for experiments. Then the corresponding eigenfaces as in Figure 10 are obtained by projecting the training images onto the face space with most significant eigenvalues. An image from the training set can be reconstructed using its eigenvectors. That is, if a projected image belongs to the training set, it could be matched by reconstructing its eigenfaces as shown in Figure 11. When a unknown image is projected onto the face space, the results may be entirely different as shown in Figure 12. That is, even though a test image is not in the training data, an image with lowest score is recognized and mapped. To overcome this anomaly, we use a certain threshold to actually classify the images.

4.4 Eigenspam

The face recognition technique using eigenvectors [37] inspires our approach to detect spam images. As images of different people have similarities corresponding to the outline of their facial features, different spam images also have some common characteristics like the wild backgrounds and the text. For the following section, we will refer to "Eigenspam" as the set of eigenvectors that constructs a spam image as

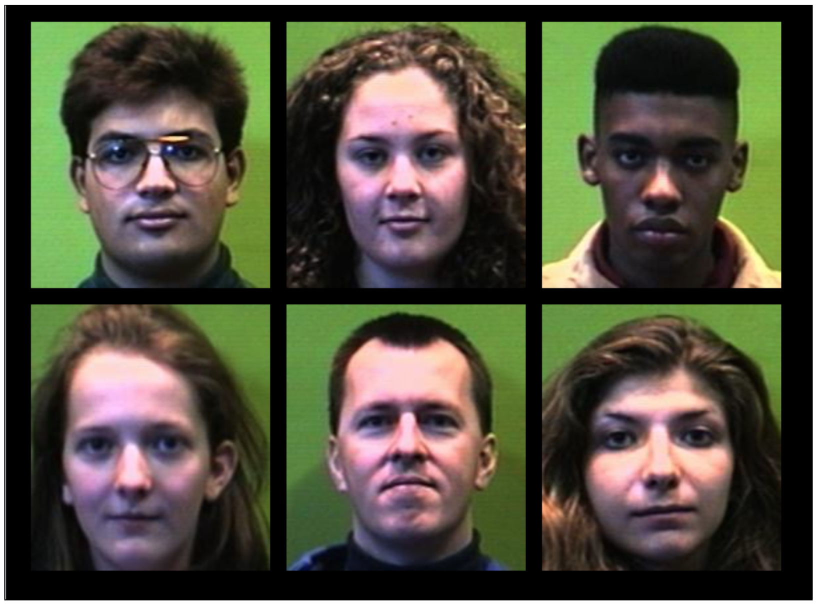


Figure 9: Facial images

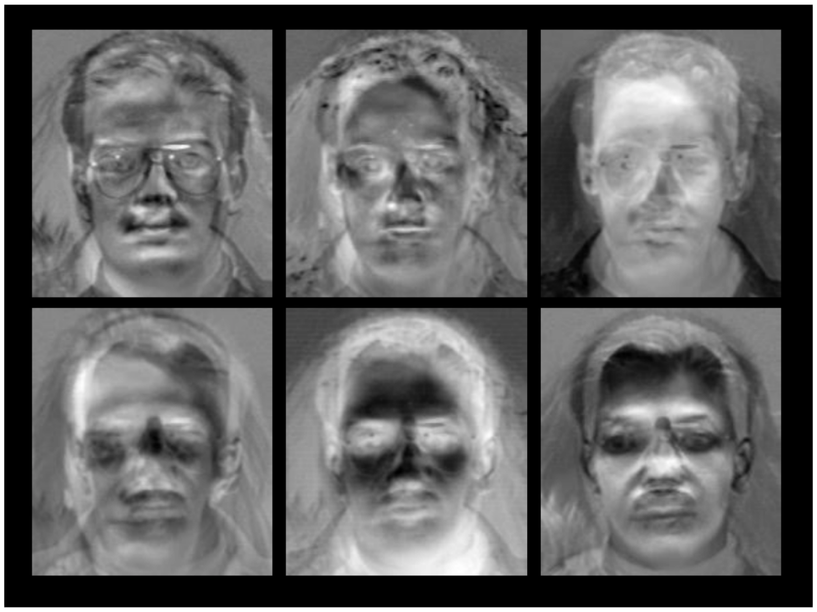


Figure 10: Eigenfaces

corresponds to Eigenfaces, which constructs the face when combined linearly. The training phase and testing phase for the system are discussed in detail below [35].

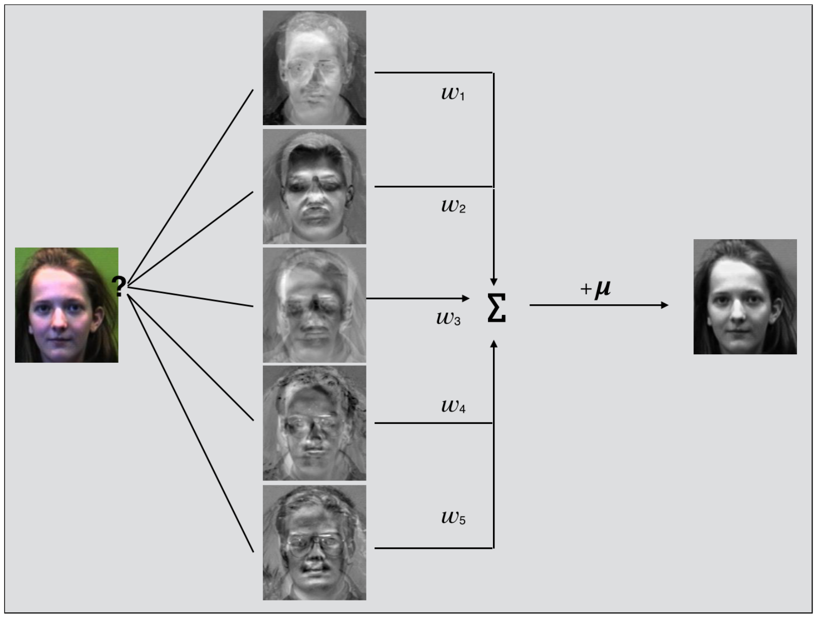


Figure 11: Known test image mapped to training set

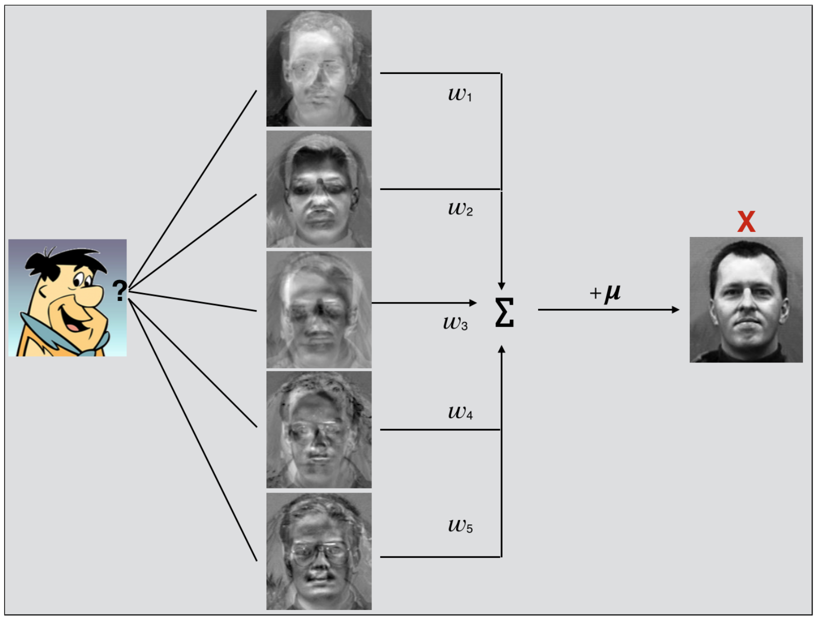


Figure 12: Unknown test image mapped to training set

4.4.1 Training Phase

The Training phase is the stage during which our model is trained by a database of different spam images called the training set. We determine the eigenvectors and project them onto the eigenspace to compute the weight vectors. The training phase consists of the following steps:

1. Acquire a set of M spam images, resize all images to be of same dimensions.

2. Convert each image to greyscale. Each image is represented as a vector V_i by converting to greyscale and flattening the image. Since all images are resized to same dimensions, each image consists of a same number of pixels. We represent the entire image set as $N \times M$, where N is the count of pixels in each image, and M is the number of images.

3. Compute the mean image vector and subtract from each image. Let $e_i(V)$ be the i^{th} element of the vector V. Then, for $j = 1, 2, 3, \ldots, N$, we let

   $$\mu_j = \frac{1}{M} \sum\nolimits_i e_j(V_i),$$

   that is, μ_j is the mean of the elements appearing in position j of the training vectors V_i. The vector of means is defined as

   $$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_N \end{pmatrix}$$

   Subtract the mean from each vector V.

   $$\tilde{V}_i = V_i - \mu$$

   The matrix of interest is defined as

   $$A = (\tilde{V}_1 \quad \tilde{V}_2 \quad \ldots \quad \tilde{V}_M) \tag{5}$$

   that is, column i of A is the vector $\tilde{V}_i$. By construction, the mean of the elements in each row of A is 0.

4. The covariance matrix [23] is

$$C = \frac{1}{M} A A^T$$

and the elements of the matrix C are denoted as c_{ij}. Then

$$c_{ij} = \frac{\tilde{V}_i \cdot \tilde{V}_j}{M} \tag{6}$$

where $\tilde{V}_i \cdot \tilde{V}_j$ denotes the dot product.

5. Find the normalized eigenvectors of covariance matrix C. However, C is $N \times N$, where N is the number of pixels of an image in the training and the number N is generally large. So, instead of instead of directly computing the eigenvectors of C, we use the following efficient approach.
Suppose a matrix L

$$L = A^T A \tag{7}$$

Here L is a $M \times M$ matrix. Since M is the number of the images in the training set, it will be much smaller than N. Let ν is an eigenvector of L, that is,

$$L\nu = \lambda\nu$$

for some λ. Then multiplying by A

$$AL\nu = AA^T A\nu = A\lambda A\nu$$

and thus, $A\nu$ is the eigenvector of C with a corresponding eigenvalue λ.

6. Suppose that the set of eigenvectors of L from the above step is $\nu = \{\nu_1, \nu_2, \ldots, \nu_j\}$ and obtain the corresponding eigenvectors of C using $\mu_i = A\nu_i$, for $i = 1, 2, \ldots, j$. Thus, consider the eigenvectors of C as $u = (u_1, u_2, \ldots, u_M)$ and sort them in descending order based on the magnitude of the eigenvalue.

7. Since we have already mentioned that the most significant eigenvectors correspond to the eigenvalues with the largest magnitude. We select the first m singular vectors of u_i where $m \leq M$ and ignore eigenvectors with small eigenvalues.

8. Project each image of the training set V onto the eigenspace [41]. We compute weight vectors Ω_i, for $i = 1, 2, \ldots, M$, as
$$\Omega_i = \begin{pmatrix} \tilde{V}_i \cdot u_1 \\ \tilde{V}_i \cdot u_2 \\ \vdots \\ \tilde{V}_i \cdot u_m \end{pmatrix} \tag{8}$$
where the dot product for the vectors $x = (x_1, x_2, \ldots, x_n)$ and $y = (y_1, y_2, \ldots, x_n)$ is defined as
$$x \cdot y = \sum_{j=1}^{n} x_j y_j.$$

9. Finally, define the scoring matrix Δ
$$\Delta = (\Omega_1, \Omega_2, \ldots, \Omega_m). \tag{9}$$
Thus, we can view Δ as a model that is trained using images in V.

4.4.2 Testing Phase

In the testing phase, a new image is projected onto the eigenspace of the training set and the Euclidean distance is computed between the test image weight vector and each of the weight vectors of the training set images. If the distance measure, also called as the score, is below a certain threshold, then the test image is classified as a spam image or it is marked as a ham image. Given an image vector X that we want to score, we proceed as follows.

1. Analogous to (8), we compute the weight vector, W of $\tilde{X}$ where $\tilde{X} = X - \mu$,

$$W = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{pmatrix} = \begin{pmatrix} \tilde{X}_i \cdot u_1 \\ \tilde{X}_i \cdot u_2 \\ \vdots \\ \tilde{X}_i \cdot u_m \end{pmatrix} \tag{10}$$

2. Compute each Euclidean distance between W from (10) and weight vector Ω_i in (9). Suppose ϵ_i is the distance between W and Ω_i, then

$$\epsilon_i = d(\Omega_i, W).$$

3. Finally, determine the score

$$\text{score}(X) = \min_i(\epsilon_i).$$

   This score(X) determines how closely the input image represents the spam image features among the training set. The smaller the score, the better is the match to the training set image. The classification of the images is based on a threshold value for the score and is set experimentally.

CHAPTER 5

Implementation

This chapters discusses the implementation of the techniques presented in the previous chapters. The first section gives implementation details of the SVM-based approach and the second section gives the implementation of PCA-based detection.

5.1 SVM-based Detection

We have implemented SVM using Python's Scikit-Learn package [30]. The support vector machines in Scikit-Learn supports both dense and sparse vectors as input. Internally it uses libsvm [4] library to handle all the computations. It also provides an option to train the classifier with different kernel functions. Figure 13 shows the internal working of our SVM-based approach for image spam detection. We discuss the training phase and testing phase executions in detail in the following sections.

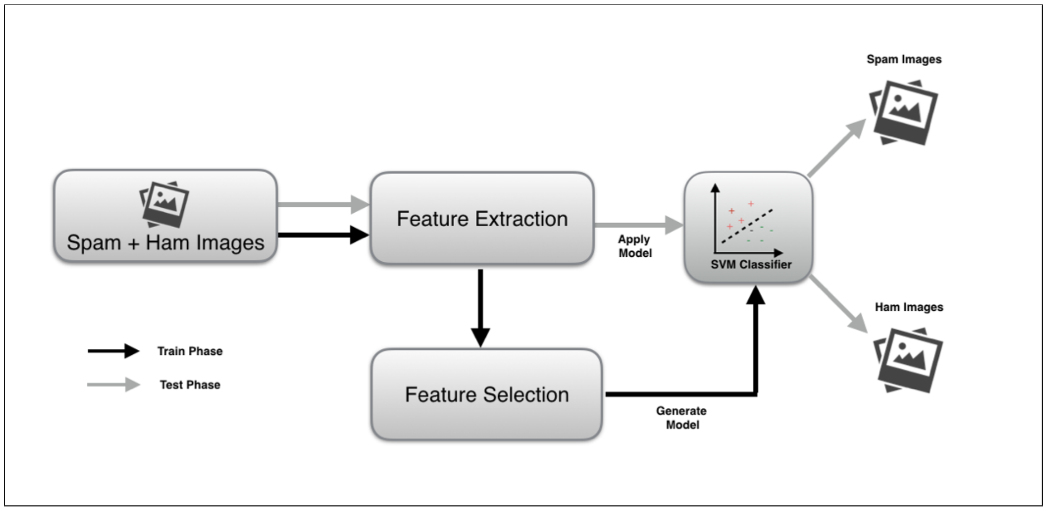


Figure 13: SVM-based detection model

5.1.1 Training Phase

Initially, we construct a training set with a combination of spam and ham images. The feature extraction module takes images as input and generates a CSV file with a list of images and the corresponding feature values. The feature selection module takes this CSV file as input to create a model and return the subset of features selected.

5.1.1.1 Feature Extraction Module

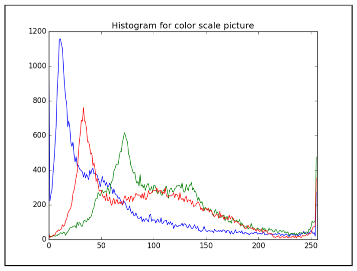
We represent the train set images as $M \times N$ matrix, where M is the number of images and N is pixel count . We extract a broad set of features for each of the train set images by manipulating the array. To obtain the entropy of color histogram, we first build a 10^3 dimension color histogram in the joint RGB space by quantizing each color band [7]. The entropy of the color histogram is then computed. We further set up one 100-dimensional histogram for each of the three color channels. Then mean, variance, skewness and kurtosis for each of the three histograms are calculated. The entropy of the LBP histogram is calculated as one texture feature. We extract 59-dimensional texture histogram, including 58 bins for all the different uniforms local binary pattern and measure the entropy value. A $40 \times 8 = 320$-dimensional gradient magnitude-orientation histogram is built to describe the shape information. The entropy of the gradient magnitude orientation histogram is the first shape feature. The total number of edges and the average length of the edges are computed by running Canny edge detector [3]. Compression ratio is computed using following:

$$\text{Compression ratio} = \frac{\text{height} \times \text{width} \times \text{bit depth}}{\text{file size}}$$

Aspect ratio is measured as the ratio between the width and height of the image. Signal to noise ratio is computed as the proportional relationship between mean and variance. The noise of the image is measured by measuring deviation at each pixel of the image.

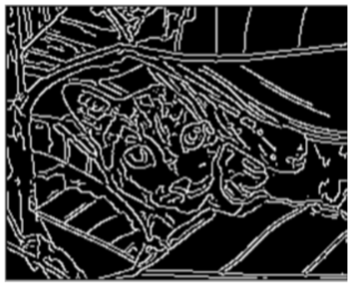
To illustrate the feature extraction, consider a picture as shown in Figure 14a. The color histogram of three different channels (RGB) are computed and plotted as shown in Figure 14b. The image is then converted to a greyscale as presented in Figure 14c. We use Canny edge detector to detect the edges. With a variation in the parameters, the edges could be detected as shown in Figure 14d and 14d. The histogram of oriented gradients (HOG) is computed and displayed as in Figure 14f. The similar illustration is also presented for a spam image in Figure 15. Finally, the entropy of these histograms is computed as feature vectors.

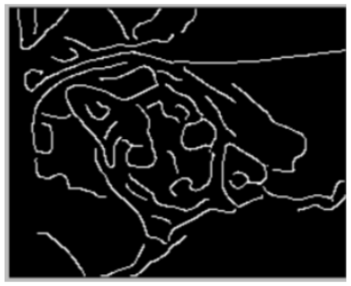
(a) Ham Image



(b) Color Histogram

(c) Greyscale


(d) Canny edges $\sigma = 1$


(e) Canny edges $\sigma = 3$

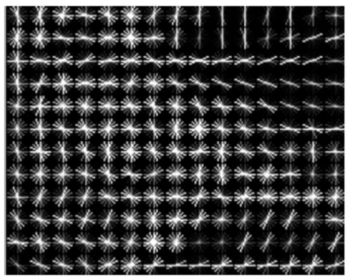
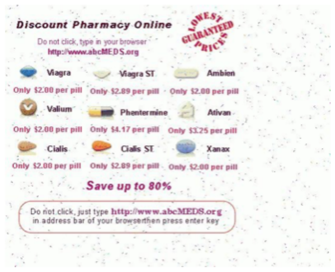
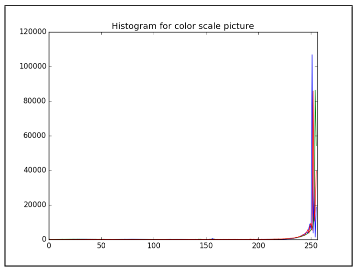
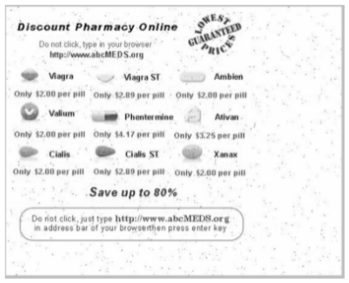

(f) HOG

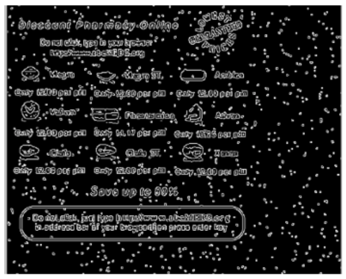
Figure 14: Feature extraction illustration of a Ham Image

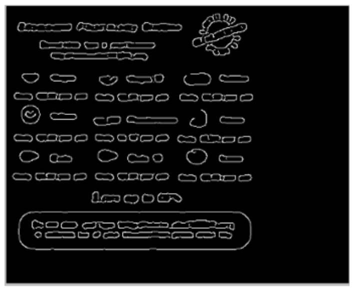

(a) Spam Image



(b) Color Histogram


(c) Greyscale


(d) Canny edges $\sigma = 1$


(e) Canny edges $\sigma = 3$

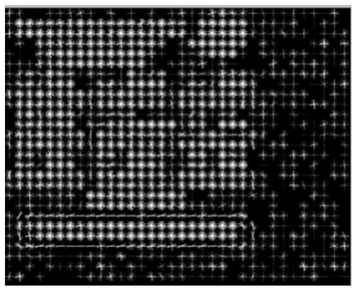

(f) HOG

Figure 15: Feature extraction illustration of a Spam Image

5.1.1.2 Feature Selection Module

The extracted features along with the class label (1 for spam and −1 for ham) are given as input to a linear SVM classifier. We generate a model after selecting a subset of features using SVM weights. Feature ranking using linear SVM weights has been implemented using the algorithm described in Table 2.

Recursive feature elimination algorithm has been implemented based on the algorithm [14] in Table 3. A model is generated using the selected features. This model is used to classify the images.

Table 2: Feature Ranking Algorithm

```
// Input: Training set $X_0 = [x_1, x_2, \ldots, x_k, \ldots, x_n]^T$,
//          Class labels $y = [y_1, y_2, \ldots, y_k, \ldots, y_n]^T$
// Output: Feature ranked list $r$
// Initializations
Set of features $S = [1, 2, \ldots, m]$,
Feature ranked list $r = [\ ]$
Max count of features to be selected $= s$
Begin
    Train the classifier
        $\lambda = \text{SVM-Train}(X, y)$
    Compute the weight vector of dimension length($S$)
        $w = \sum_k \lambda_k y_k x_k$
    Compute the ranking criteria
        $c_i = (w_i)^2$, for all $i$
    Sort the features based on weights
        $f = \text{sort}(c)$
    Update feature ranked list
        $r = [S(f), r]$
    Select the first $s$ features
        $S = S(s)$
Return $r$, $S$
End
```

5.1.2 Testing Phase

A test set of spam and ham images is given as input to the feature extraction module to extract the selected subset of features. Then the extracted features are classified using SVM algorithm.

Table 3: Recursive Feature Elimination Algorithm [14]

```
// Input: Training set $X_0 = [x_1, x_2, \ldots, x_k, \ldots, x_n]^T$,
//          Class labels $y = [y_1, y_2, \ldots, y_k, \ldots, y_n]^T$
// Output: Feature ranked list $r$
// Initializations
Subset of surviving features $S = [1, 2, \ldots, m]$,
Feature ranked list $r = [\ ]$
Begin
Repeat until $S = [\ ]$
    Constrain training $X$ to feature set $S$
        $X = X_0(:, S)$
    Train the classifier
        $\lambda = \text{SVM-Train}(X, y)$
    Compute the weight vector of dimension length$(S)$
        $w = \sum_k \lambda_k y_k x_k$
    Compute the ranking criteria
        $c_i = (w_i)^2$, for all $i$
    Find feature with smallest ranking criterion
        $f = \text{argmin}(c)$
    Update feature ranked list
        $r = [S(f), r]$
    Discard feature with smallest rank
        $S = S(1 : f - 1, f + 1 : \text{length}(S))$
Return $r$
End
```

5.2 PCA-based Detection

We implemented the algorithm discussed in Chapter 4 using Python. Python Imaging Library (PIL) and OpenCV packages have been used to work with the images. Figure 16 summarizes the eigenspam detection process.

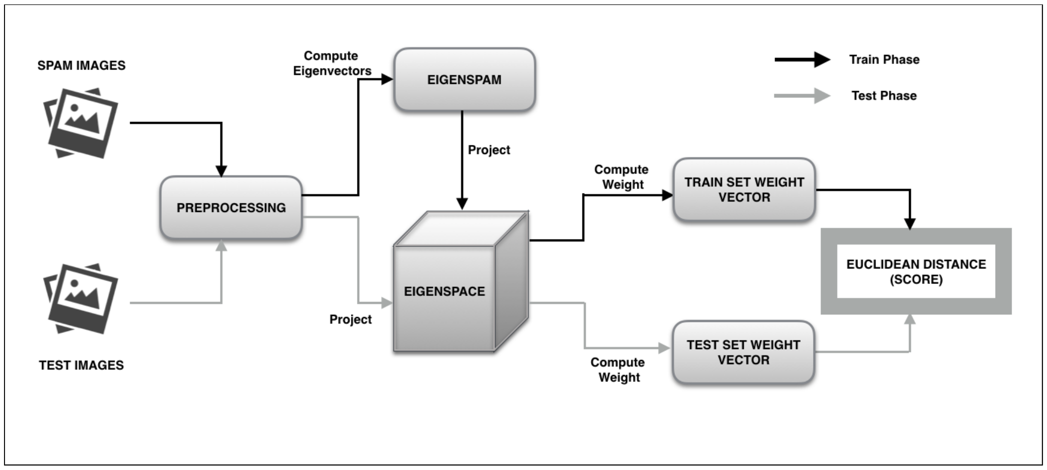


Figure 16: Eigenspam based detection model

5.2.1 Training Phase

We construct a test set of spam images. Initially, we resize all the images and convert them to grayscale. Converting to grayscale, flattens the image and allows us to store in a matrix $A_{M\times N}$, where N is total pixel count of an image and M is a total number of image files used for training. This matrix is given as an input to compute eigenvectors and weights. Python's Scikit-Learn has been used to implement this algorithm. We store the weights for further manipulations.

5.2.2 Testing Phase

Test set constitutes both spam and ham images. Project each image from the test set onto the eigenspace created in the training phase and compute the weight vector. Then, we measure the Euclidean distance from each of the weight vectors of the training set. We classify the derived score based on an empirical threshold value.

The experiments for the above implementations have been presented in Chapter 6.

CHAPTER 6
Experimental Analysis and Results

In this chapter, we present the empirical analysis and the results of our project. The first section gives details about the datasets used and created by us, followed by the experimental setup for the project. We discuss the criteria for the evaluation in the third section. The rest of the chapter shows the experiments and results.

6.1 Datasets

Very few image spam corpus are available to the public due to privacy concerns. We analyze the project using such available datasets.

6.1.1 Dataset 1

This dataset is available at Northwestern University's public website [13]. This dataset was created by authors of Image spam hunter [12] by collecting images from the original emails. The dataset consists of a total of 928 spam images and 810 ham images. The dataset consists of images in JPEG format. We excluded eight images from the spam data as they were corrupted. The dataset has been divided into two parts as training set and test set.

6.1.2 Dataset 2

This data was used by the authors of Learning Fast classifiers for Image spam [7] and was made accessible to public [8]. The data set consists of a large corpus with different formats (JPEG, GIF, PNG), most of which have been found either repeated or corrupted. So, we selected a subset of 1091 spam images and 1056 ham images and converted them to JPEG format for our experiments.

6.1.3 Dataset 3

We created a new set of spam images as mentioned in Chapter 3. The idea is to build a stronger set that would bypass the existing spam filters. As the current filters rely on the highly determinant characteristics of the spam, we have processed the images to make those visible properties similar to the properties of a ham image. We generated a set of 1029 such images and performed various experiments to test the sanity of the data.

In addition to the above datasets, we have also performed our experiments on the Princeton spam image benchmark dataset [28].

6.2 Experimental Setup

This project has been implemented using Python-3.5. Image processing packages such as Scikit-Image, Python Imaging Library (PIL) and OpenCV have been used to work with images. Scikit-Learn package has been used to implement machine learning algorithms for the detection. Matplotlib has been used to plot the graphs for the analysis. We worked with Mac OS X operating system.

6.3 Evaluation Criteria

The proposed techniques have been evaluated based on accuracy, false positive rate, and area under the curve. The accuracy of a classification is the ratio of a number of correctly classified samples to the total number of samples [35]. We define accuracy as

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative}}$$

Where, True Positive is the number of spam images correctly classified as spam, True Negative is the number of ham images correctly classified as ham, False Positive is the number of ham images classified as spam and False Negative is the number of spam images incorrectly classified as ham images. False Positive Rate (FPR) is the percentage of the genuine files that incorrectly fall into the spam class. FPR for a classification scheme is

$$\text{FPR} = \frac{\text{False Positive}}{\text{True Positive} + \text{False Positive}}.$$

In machine learning, Receiving Operating Characteristics (ROC) is a graphical plot used to compute the Area Under the Curve (AUC) value and determine the efficiency of the algorithm. We generate the ROC curve by plotting the True Positive Rate (TPR) versus the False Positive Rate (FPR) with varying threshold through the range of data points. Figure 17 shows ROC curve with a shaded region. We compute the area of the shaded region to obtain the AUC value. This value usually lies in between 0.5 to 1. An AUC value of 1 indicates ideal classification with zero false positives or false negatives.

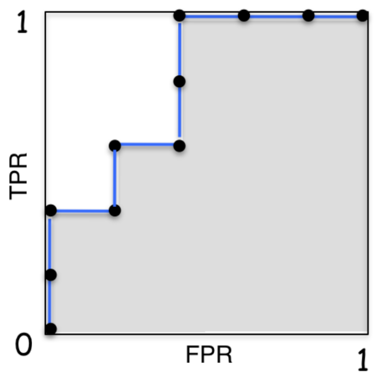


Figure 17: ROC curve with shaded area

In this project, we evaluate the performance of the proposed techniques majorly by plotting ROC curve and calculating the AUC value.

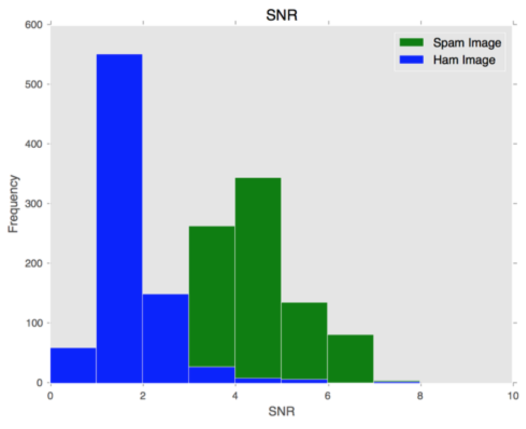
6.4 SVM Model Experiments

6.4.1 Image Feature Statistics

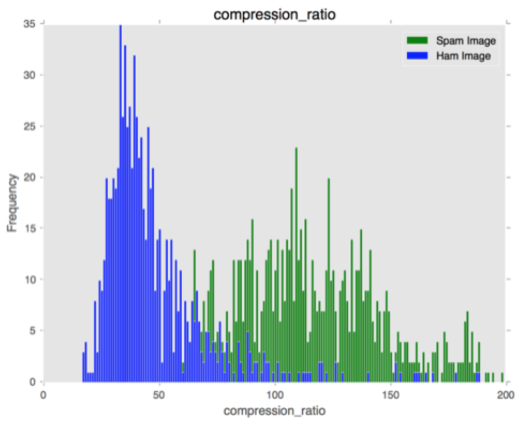
The initial step of this method is to retrieve 21 image features. These features are adopted with the motivation that the spam image possesses different visual statistics compared to the genuine image. To demonstrate this, we first plotted the distributions for each of the 21 features in two distinct classes for dataset 1. The visualizations of the most effective features have been presented in Figure 18.

We can observe that the chosen features can certainly distinguish between the spam image and ham image. There is a clear mark of separation between the two classes of images. Further, we plotted the ROC curves to compute the AUC value for each of the features score obtained using SVM classifier. The ROC curves corresponding to the features in Figure 18 can be observed in Figure 19. It is apparent from the results that each feature by itself discriminates strongly between both the classes. The distributions and the ROC graphs for all the features are shown in the Appendix A.

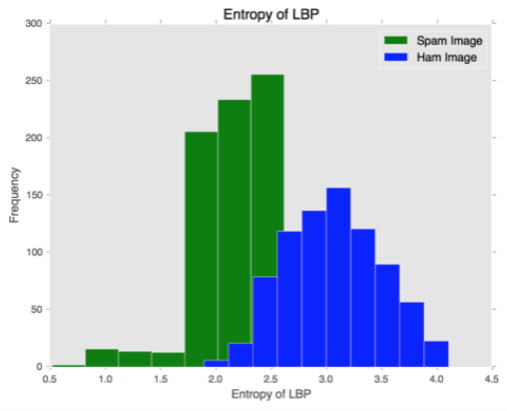
Further we plot the AUC for all the 21 features for images from dataset 1 as shown in the Figure 20. We observe that the variances of different color channels produce a high classification rate. The color histogram based statistical features are most active in the exposition of the spam images. The analysis of the plot strengthens the fact that the spam image possesses properties that could be distinguished from the normal images. And, each feature individually could classify the images to a reasonable degree.



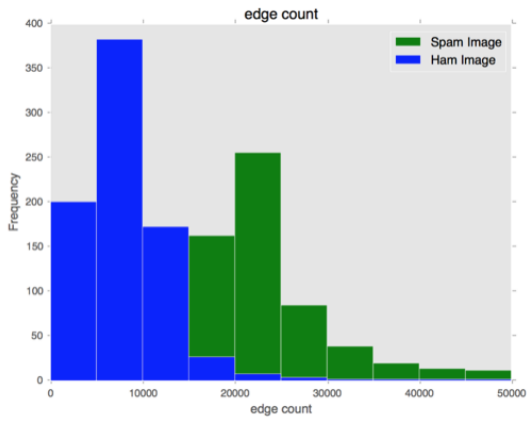
(a) Signal to noise ratio



(b) Compression ratio



(c) LBP

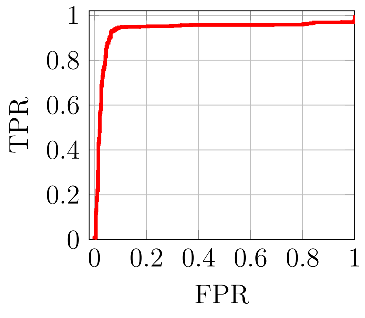


(d) Edge Count

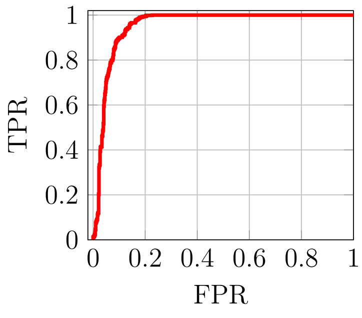
Figure 18: Feature distribution for spam versus ham images

6.4.1.1 Kullback-Leibler Divergence

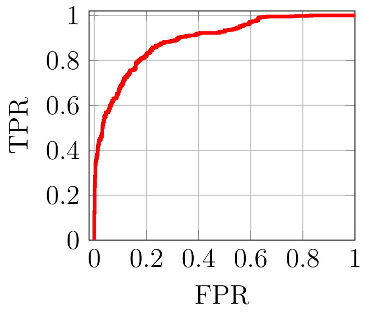
We further computed the Kullback-Leibler divergence [21](known as KL divergence) for the probability distributions between the two classes of data for each feature. KL divergence is a distance metrics that measures the difference between two probability distributions M and N. Concretely, the KL divergence of N from M is denoted by $D_{KL}(M \parallel N)$ and measures the amount of added information needed to encode image M based on the histogram of image N. The KL divergence of N from



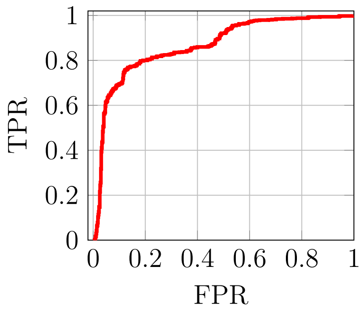
(a) Signal to noise ratio



(b) Compression ratio



(c) LBP



(d) Edge Count

Figure 19: ROC curves for the feature distributions in Figure 18

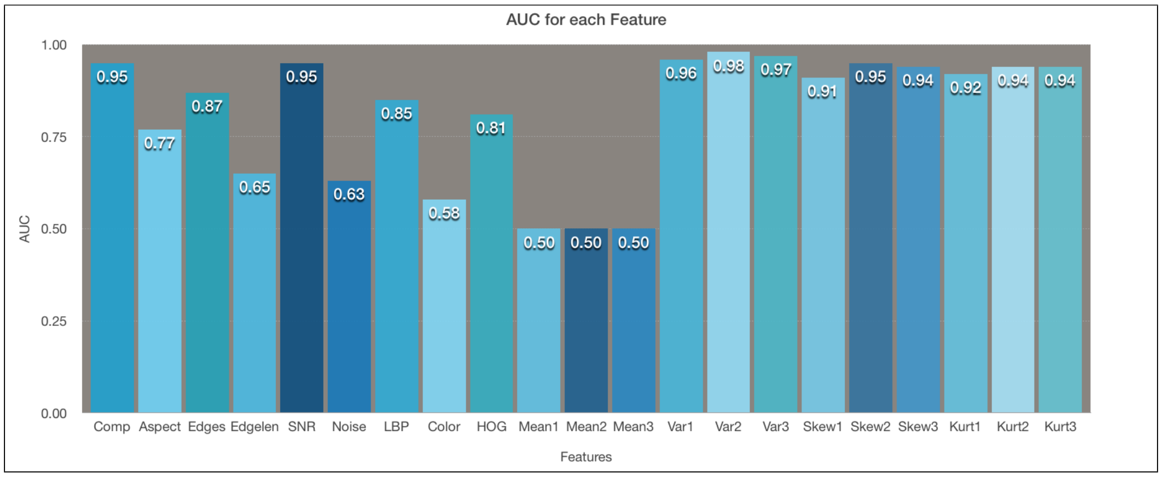


Figure 20: AUC for Feature distributions

M is given by

$$D_{KL}(M \parallel N) = \sum_i M(i) \log \frac{M(i)}{N(i)}, \tag{11}$$

where M and N are discrete probability distributions [27]. KL divergence is non-symmetric, that is $D(M \parallel N)$ is not equal to $D(N \parallel M)$.

Using equation (11), we computed the KL divergence for the feature distributions from ham to spam and spam to ham. The results are given in the Table 4.

Table 4: KL divergence for Feature distributions

Feature	**Spam to Ham**	**Ham to Spam**	**SVM AUC**
Color hist	0.01	0.01	0.58
Mean1	0	0	0.5
Mean2	0	0	0.5
Mean3	0	0	0.5
Var1	0.08	0.07	0.96
Var2	0.11	0.09	0.98
Var3	0.1	0.09	0.97
Skew1	0.23	0.23	0.91
Skew2	0.26	0.27	0.95
Skew3	0.22	0.28	0.94
Kurt1	0.32	0.29	0.92
Kurt2	0.35	0.33	0.94
Kurt3	0.31	0.36	0.94
LBP	0.01	0.01	0.85
HOG	0.04	0.03	0.81
Edge no	0.04	0.05	0.87
Edge len	0.02	0.02	0.65
Comp	0.04	0.01	0.95
Aspect	0.03	0.04	0.77
SNR	0.15	0.04	0.95
Noise	0.02	0.01	0.63

6.4.2 SVM Results without Feature Selection

We classify the extracted features using SVM technique. First, we label the two classes of images (1 for spam, -1 for ham). The classification has been performed on the first two datasets.

6.4.2.1 Dataset 1

For dataset 1, we considered the training set with 100 spam images and 100 ham images. There is no overlap between the test and the train sets. The test set contains 710 ham and 820 spam images. The model has been trained and tested with different kernel functions. The accuracy and the FPR have been computed for the classification using three kernels. The results are shown in the Table 5.

Table 5: Accuracy and FPR with different Kernel functions for dataset 1

Kernel	**Accuracy**	**FPR**
Linear	0.96	0.05
RBF	0.96	0.05
Polynomial	0.92	0.02

It is observed that the Linear kernel performed a better classification with a low positive rate compared to the other kernels.The performance of the linear SVM classification is described as a confusion matrix in Table 6.

Table 6: Linear SVM Classification for dataset 1

Actual	**Classified**	
	Spam	Ham
Spam(820)	801	19
Ham (710)	32	678

Figure 21 shows the ROC curve for the SVM classification. The corresponding AUC value for the curve is 0.99 which implies that the SVM technique detects spam images with a good accuracy and low false positive rate.

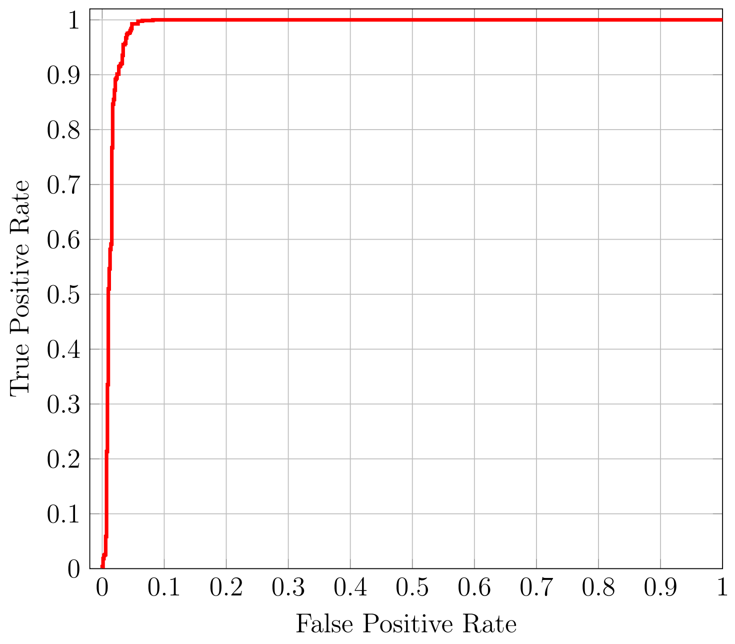


Figure 21: ROC Curve for SVM classification without feature selection for dataset 1 with AUC = 0.99

6.4.2.2 Dataset 2

Dataset 2 contains 1091 spam images and 1056 ham images. The training set is constructed with 1037 images from both classes. There is no overlap between the test and the train sets. The test set contains 800 images. The model has been trained and tested with different kernel functions. The accuracy and the FPR have been computed for the classification using three kernels. The results are shown in the Table 7.

In this case, it is observed that the Radial Basis Function performed a better classification with a low positive rate compared to the other kernels.The performance of the RBF SVM classification is described as a confusion matrix in Table 8.

Table 7: Accuracy and FPR with different Kernel functions for dataset 2

Kernel	**Accuracy**	**FPR**
Linear	0.97	0.02
RBF	0.98	0.02
Polynomial	0.96	0.07

Table 8: RBF SVM Classification for dataset 2

Actual	**Classified**	
	Spam	Ham
Spam(400)	393	7
Ham (400)	6	394

Figure 22 shows the ROC curve for the SVM classification. The corresponding AUC value for the curve is 1.0 which implies that the SVM technique detects spam images as desired.

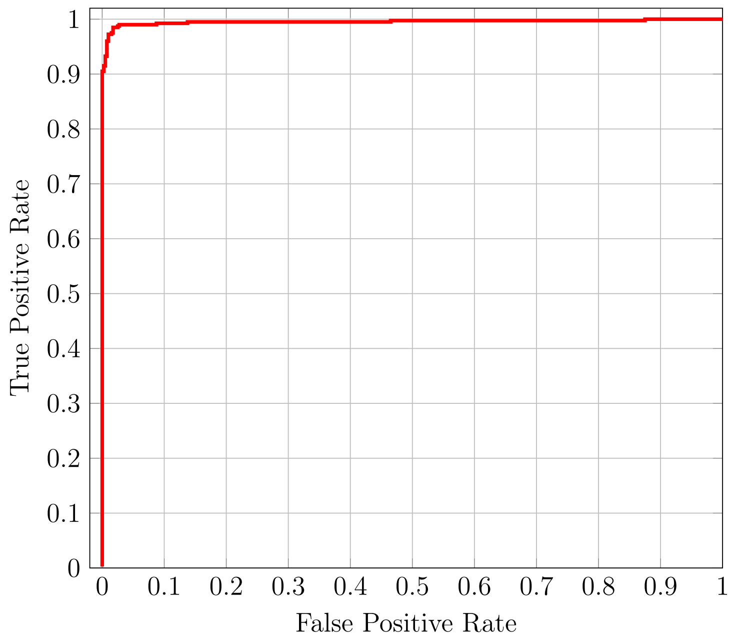


Figure 22: ROC Curve for SVM classification without feature selection for dataset 2 with AUC =1.0

6.4.3 SVM Results with Feature Selection

In this section, we perform SVM classification after selecting highly discriminant subset of features by training the model with the weights extracted for 21 features. We aim to find an optimal subset of the features to reduce the computational complexity and yet provide maximum accuracy for classification.

6.4.3.1 Dataset 1

As mentioned in the Section 5.1, we first train a Linear SVM model with 21 features extracted from the training set images. Initially, we compute the weights of the features after creating a model. Using recursive feature elimination, we select a subset of features based on the ranking criteria as mentioned under Section 5.1. That is, to choose a set of 10 elements using RFE algorithm, we compute weights and recursively eliminate 11 features. To find an optimal subset, we experimented with incrementing values for N and measured the respective AUC values based on SVM classification. Figure 23 shows the number of selected features with their corresponding AUC values and accuracy. In the plot, the orange line denotes the AUC value, and the blue bars show the accuracy for the selected number of features. We can observe that the maximum AUC value is first attained for a subset of 3 features. But, this subset cannot be chosen as optimal because its accuracy value is less than 0.96 which is not the maximum value. And as we increase the number of features in the subset, we achieve a higher accuracy. Thus, we consider 13 feature subset as an optimal set considering this data.

We compare the result with simple feature ranking approach. First, we arrange the features based on the weights. We select a subset of N features by eliminating the elements greater than rank N. That is, a subset of 10 would contain all the

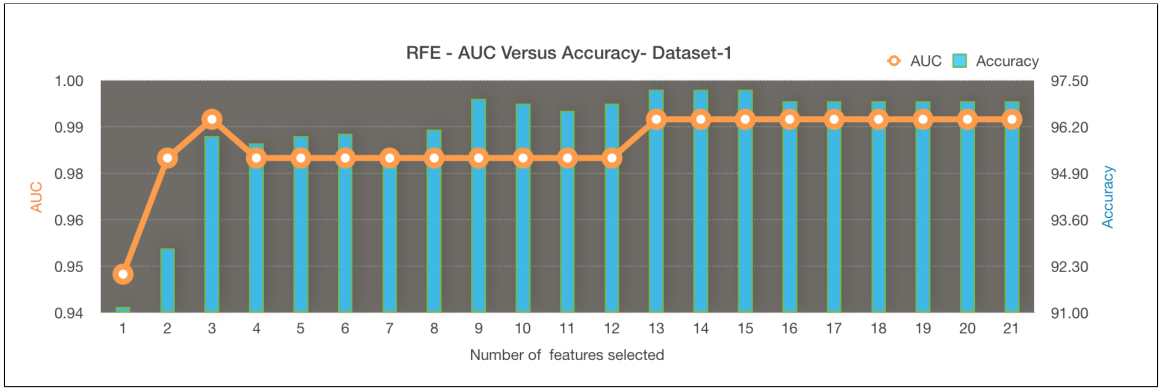


Figure 23: RFE selection - AUC versus accuracy for dataset 1

features till rank 10. We experimented with incrementing values for N and performed classification to compute the respective AUC values. Figure 24 shows the plotting for the number of features selected with ranking and its corresponding AUC with an orange line in comparison with AUC values obtained using RFE approach. Both the methods produce the maximum AUC at 13 feature subset.

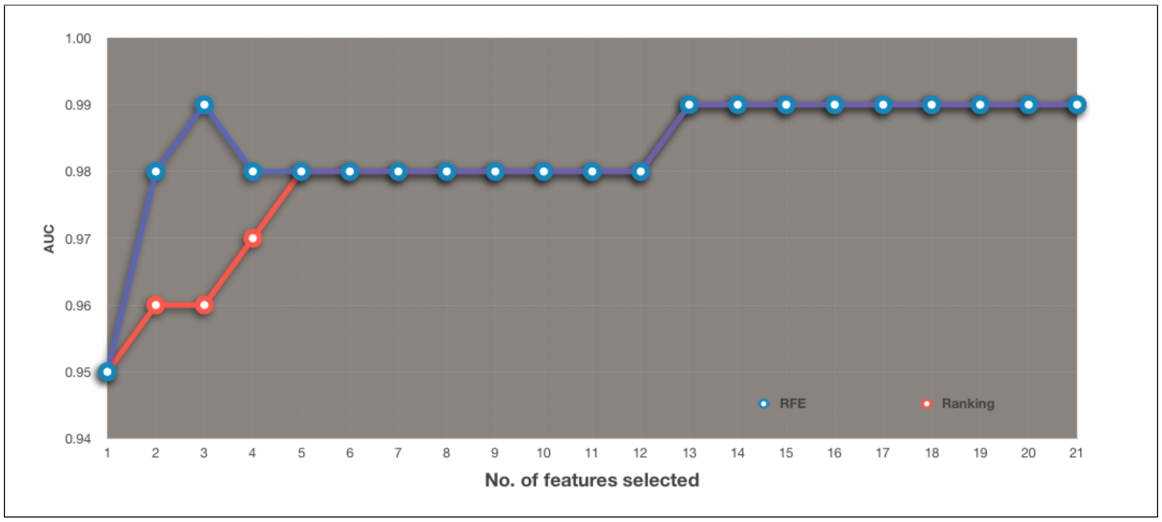


Figure 24: AUC - RFE versus ranking for dataset 1

One interesting observation from Figure 24 is the difference in AUC for selection of 3 features using both approaches. It is apparent that the combinations of the features make a difference in the result. The combination for $N = 3$ using the ranking approach from Figure 25 is the compression ratio, signal to noise ratio and entropy of HOG (numbered as 1,2,3) whereas using RFE we get compression ratio, the entropy of LBP, the variance for one color channel as in Figure 25 with blue marked bars. Thus, using RFE approach the selection of features yields maximum efficiency.

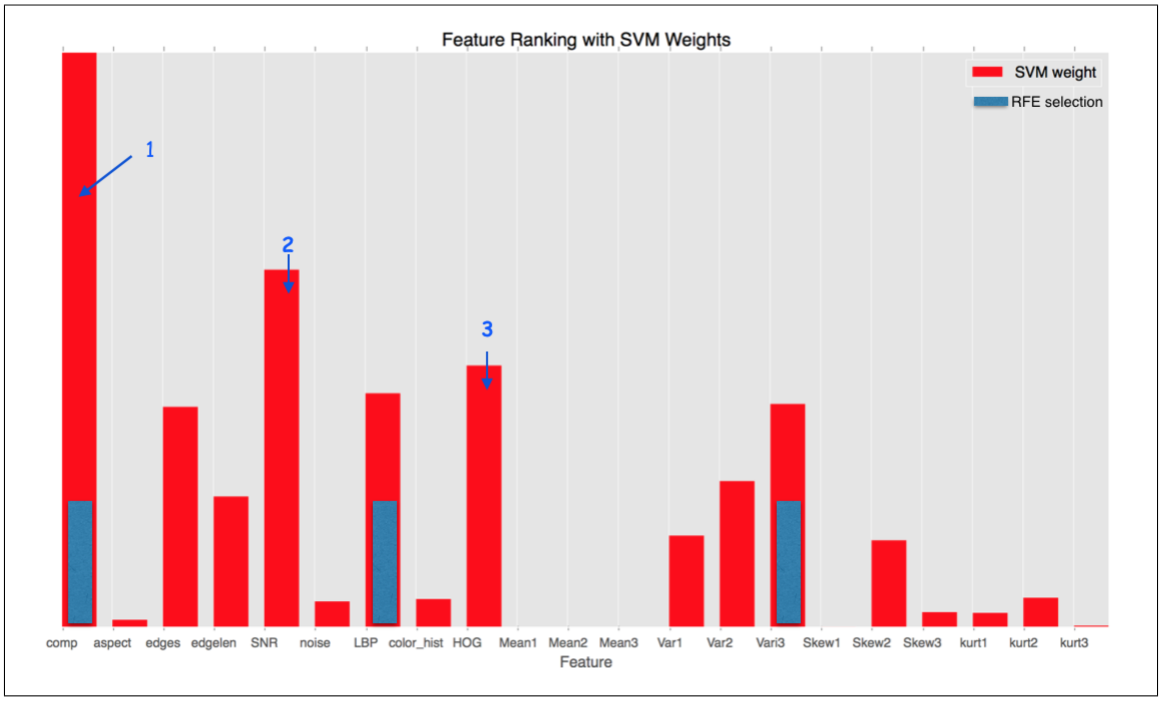


Figure 25: SVM weights ranking with RFE selection in blue bars for dataset 1

Finally, the Table 9 gives the detailed result using 13 features. Table 10 shows the accuracy of classification using different kernels of SVM. The corresponding ROC curve for the detection using 13 features is presented in Figure 26. The AUC value for the curve is computed to be 0.99.

From the results, it is also apparent that using 13 features; we get the same AUC as the classification with all the 21 features and also with a greater value of accuracy compared to accuracy without feature selection which is 0.96.

Table 9: Accuracy and FPR with different Kernel functions for dataset 1 using 13 features

Kernel	**Accuracy**	**FPR**
Linear	0.97	0.04
RBF	0.96	0.05
Polynomial	0.95	0.03

Table 10: Linear SVM Classification for dataset 1 with 13 features

Actual	**Classified**	
	Spam	Ham
Spam(820)	811	9
Ham (710)	33	677

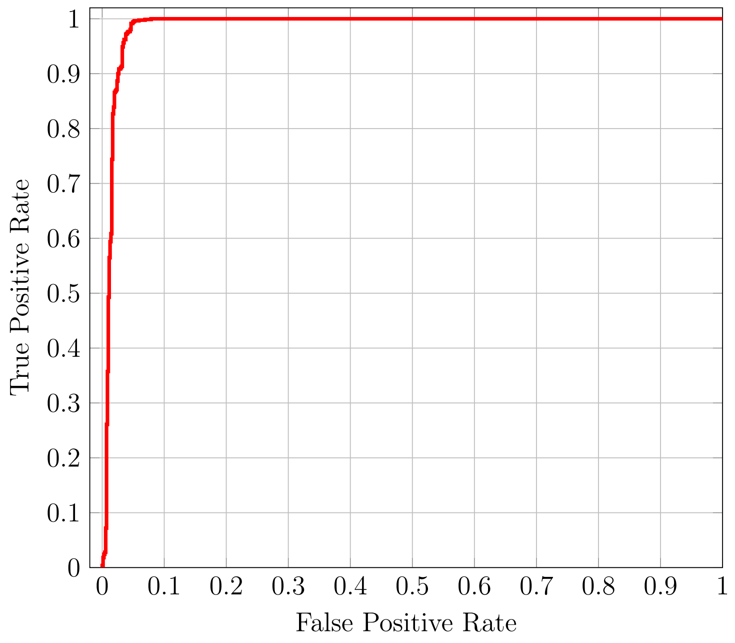


Figure 26: ROC Curve for SVM classification with RFE feature selection for dataset 1 with AUC =0.99

6.4.3.2 Dataset 2

We perform similar experiments to calculate AUC and accuracy for an increasing number of features using RFE. From Figure 27, we observe that the maximum accuracy and AUC values are seen for $N = 12$. Though maximumAUC value is obtained at $N = 9$, at $N = 12$, the accuracy is also maximum. Thus, we consider $N = 12$ as the optimum feature set for this dataset.

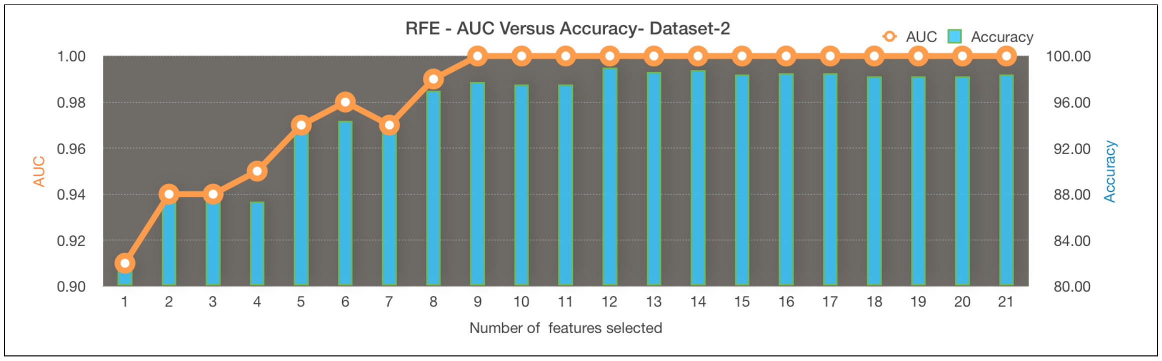


Figure 27: RFE selection - AUC versus accuracy for dataset 2

The results in Table 11 and 12 show that accuracy achieved with feature section is greater than without selection The corresponding ROC is plotted in Figure 28.

Table 11: Accuracy and FPR with different Kernel functions for dataset 2 using 12 features

Kernel	**Accuracy**	**FPR**
Linear	0.96	0.06
RBF	0.99	0.01
Polynomial	0.95	0.07

Table 12: SVM Classification for dataset 2 with 12 features

Actual	**Classified**	
	Spam	Ham
Spam(400)	398	6
Ham (400)	2	394

The comparison of AUC values for both the datasets is shown in Figure 29. From the results of both the datasets, it can be observed that this feature selection depends on the training model. The selection of the features employs a sensible approach based on the discriminant properties of the training set.

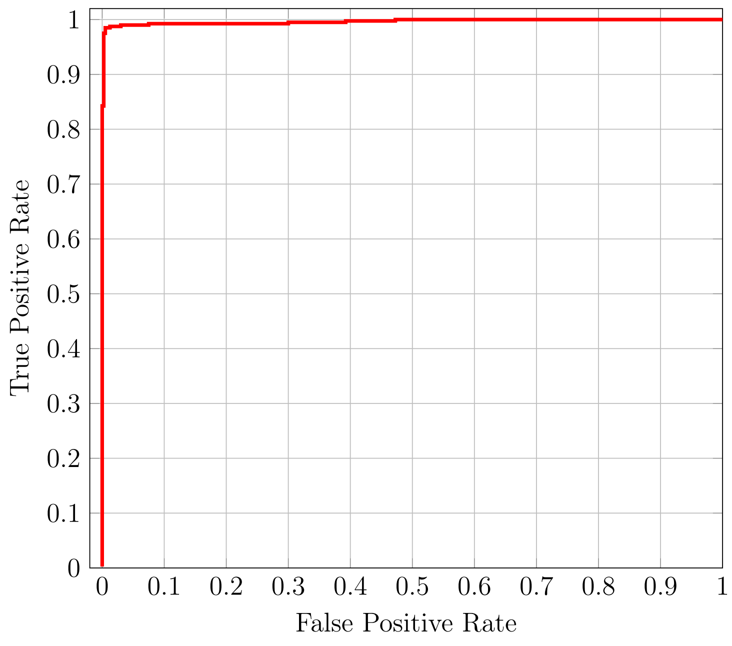


Figure 28: ROC Curve for SVM classification with RFE feature selection for dataset 2 with AUC =1.0

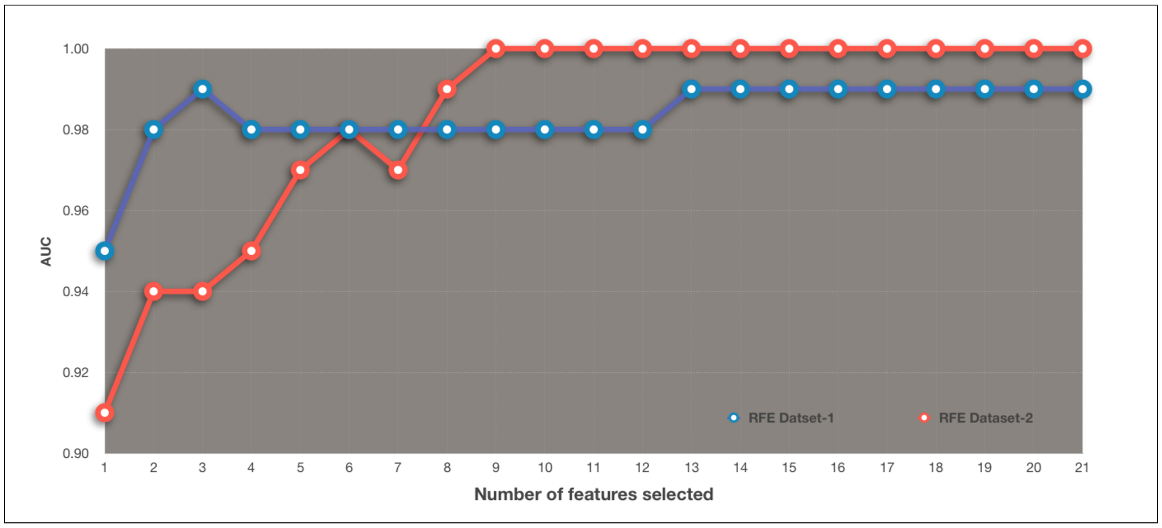


Figure 29: AUC comparison of dataset 1 and dataset 2 using feature selection

6.5 PCA Model Experiments

We perform various experiments for the first two datasets discussed in Section 6.1. We first construct a train set with spam images and a test set including both spam and ham images. The results of the Eigen spam technique are shown in the following sub-sections.

6.5.1 Eigenspam projections

Considering dataset 1, we constructed a training set with 500 spam images. The train set is projected onto the eigenspace as discussed earlier. Figure 30 shows a sample of spam images from the train set that are projected and Figure 31 shows the corresponding eigenvectors.

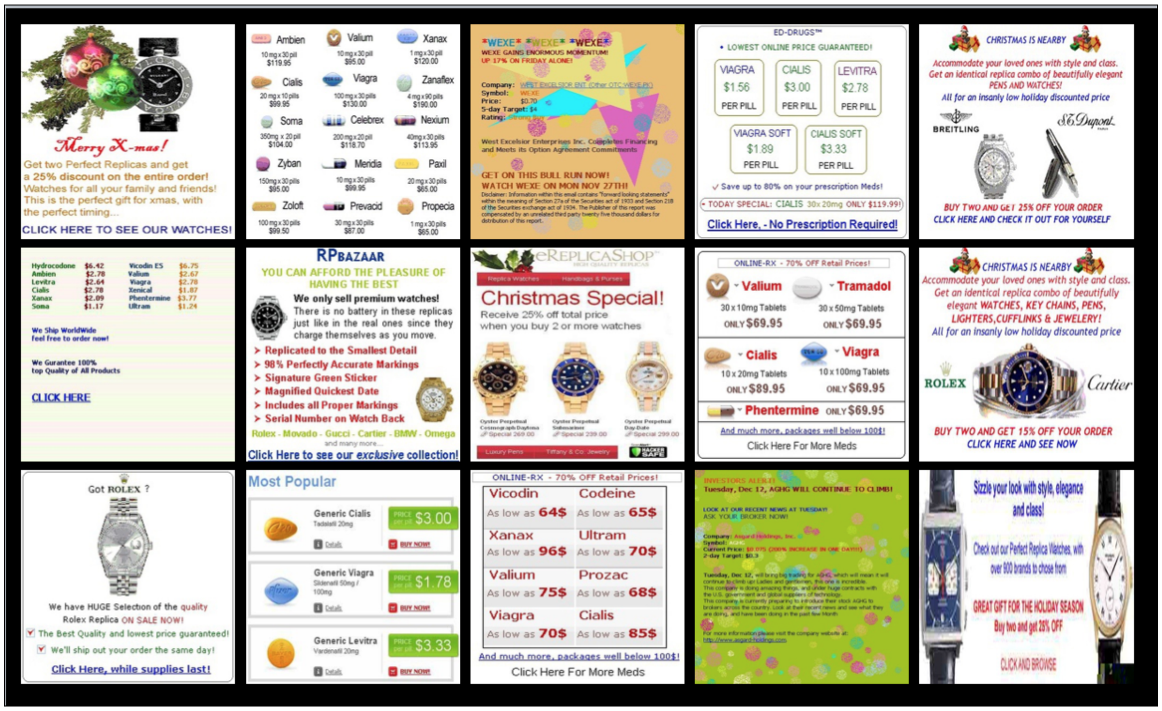


Figure 30: Spam Images from Training set

6.5.2 Results

The following sub-sections display the graphs for two different datasets. We plotted the score by computing the Euclidean distance between each of the test image and spam image from the train set. The score closer to zero indicates that the test

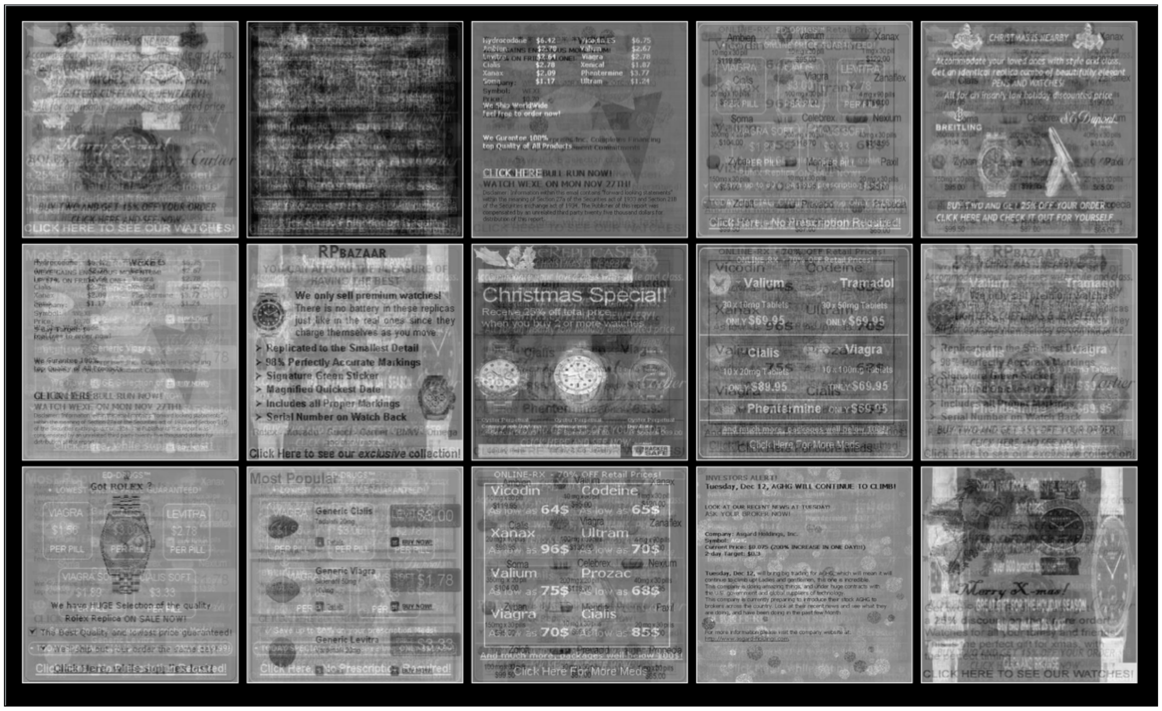


Figure 31: Eigenspam - Eigenvectors corresponding to Images in Figure 30

file is very similar to the training set spam image.

6.5.2.1 Dataset 1 Results

We considered 500 spam images as the train set, and the test set is a combination of 414 images from each class. Figure 32 displays the scatter plot for the scores considering 5 Eigen components. The accuracy of the classification is computed as 0.97.

Figure 33 shows the ROC plotting for the scatter plot in the Figure 31. The corresponding AUC value is 0.99 which implies that most of the spam images are successfully classified considering its different forms.

We also computed the scores for increasing the count of Eigen components from 1 to 500. The corresponding AUC values have also been computed. Figure 34 shows

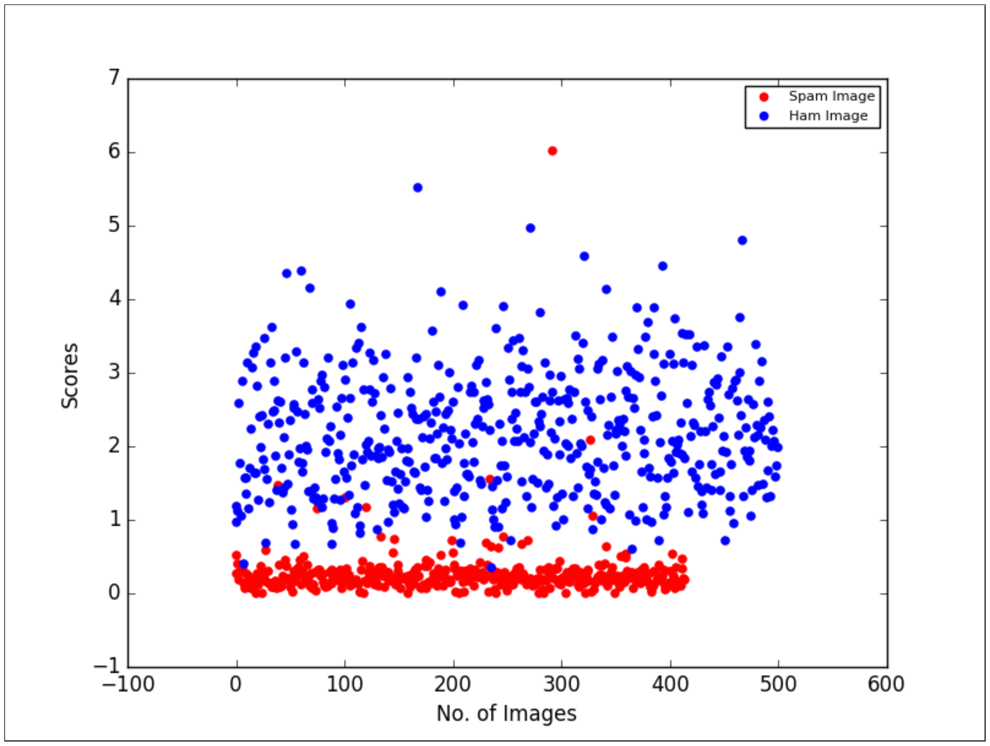


Figure 32: Scatterplot of score for dataset 1

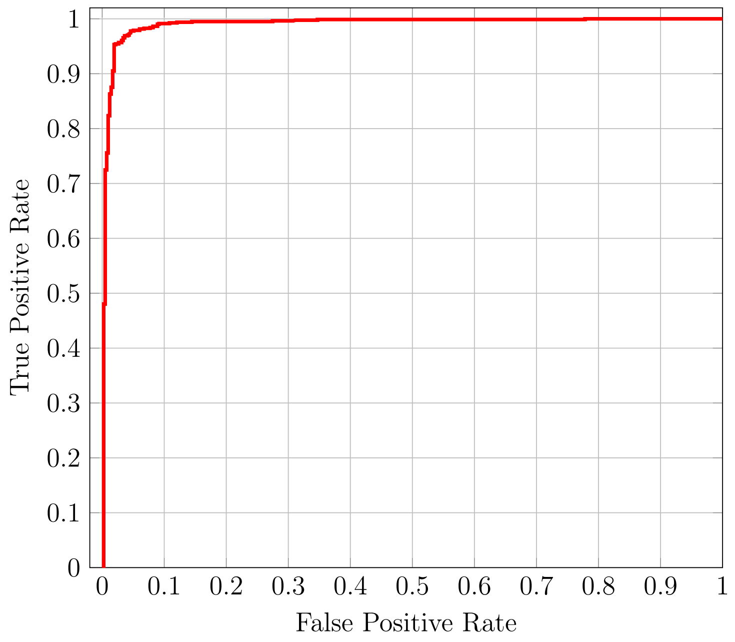


Figure 33: ROC Curve for Eigenspam with AUC = 0.99 for dataset 1

the plot for the number of Eigen components with respective AUC values. We observe that the maximum efficiency is achieved when the number of Eigen components is in between 3 to 10 and decreases gradually.

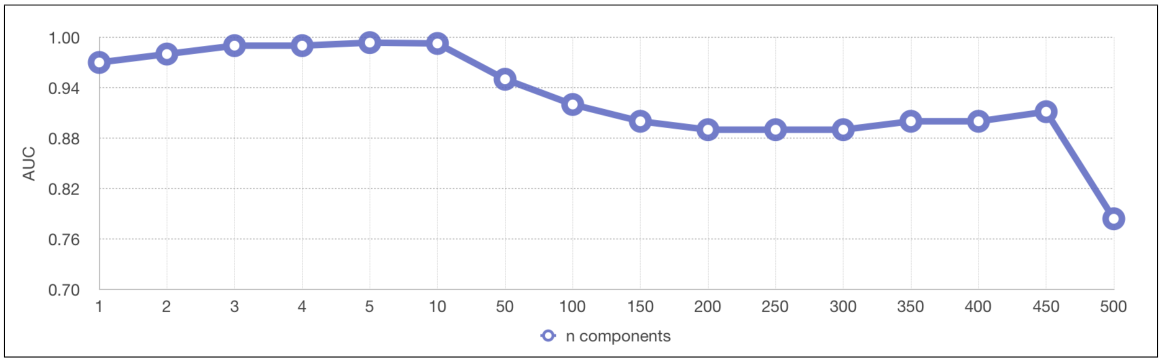


Figure 34: Eigen components versus AUC for dataset 1

6.5.2.2 Dataset 2 results

We performed experiments with 789 spam images as training set and with 300 images from each class as the test set. Figure 35 displays the scatter plot for the scores considering 5 Eigen components. Figure 36 shows the ROC plotting for the scatter plot in the Figure 35. The corresponding AUC value is 0.99 which implies a relatively perfect classification.

We also computed the scores for increasing the count of Eigen components from 1 to maximum and the corresponding AUC values. Figure 37 shows the plot for the number of Eigen components with respective AUC values. We observe that the maximum efficiency is achieved when the number of Eigen components is equal to three and decreases gradually.

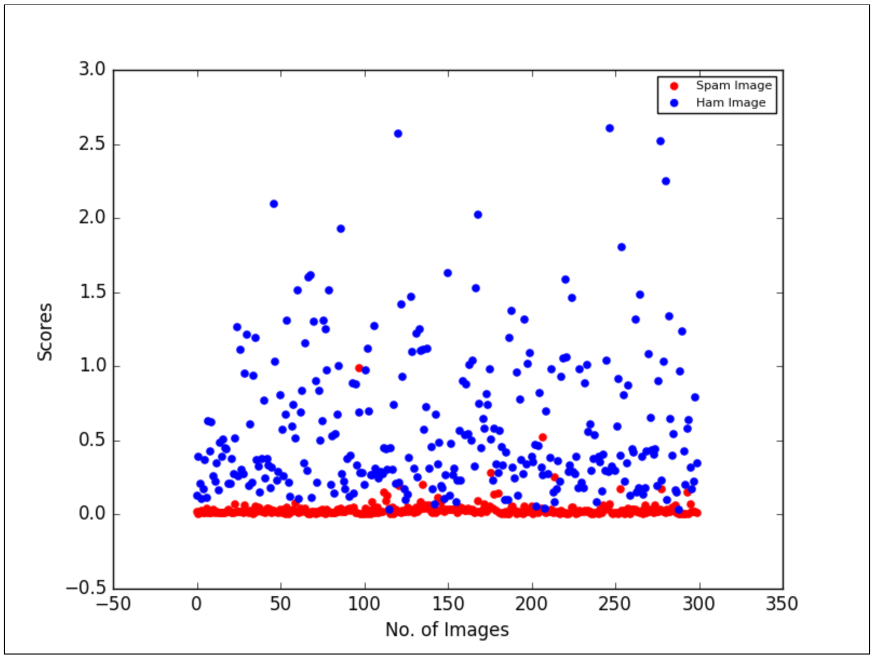


Figure 35: Scatterplot of score for dataset 2

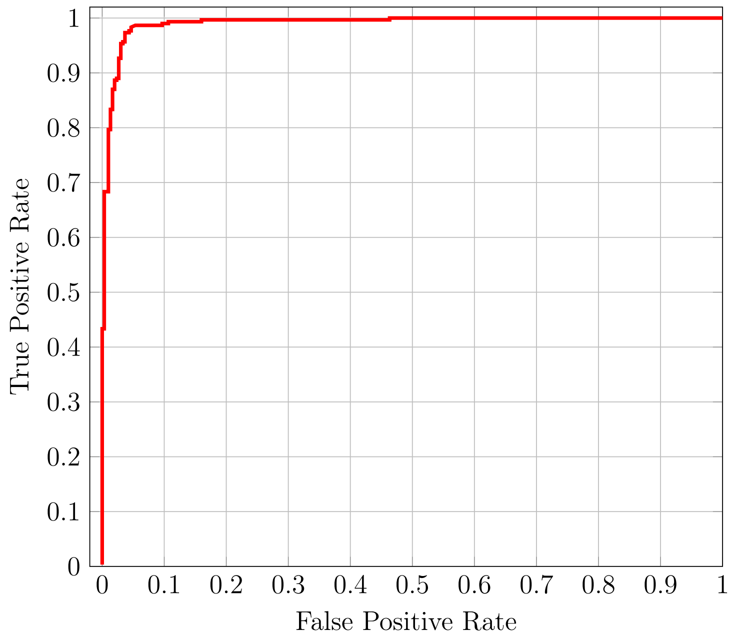


Figure 36: ROC Curve for Eigenspam with AUC = 0.99 for dataset 2

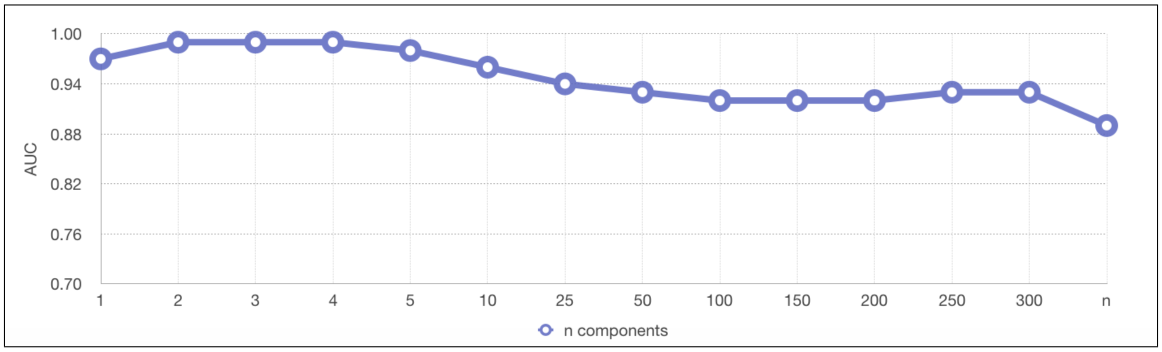


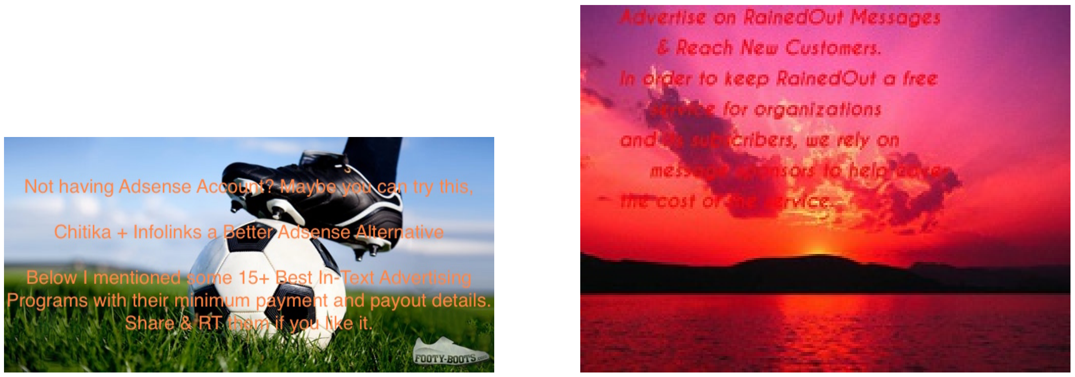
Figure 37: Eigen components versus AUC for dataset 2

6.6 New Dataset Creation and Experiments

Recall that in the feature selection model; we discover the most discriminant features of the spam images using the weight vector. Using the results from dataset 1 and dataset 2, we could analyze that the most significant features are the compression ratio, the variance of the color histogram, signal to noise ratio, and the local binary pattern. The fact that these visible characteristics of the spam image are the base for the filtering criteria motivated us to create a stronger spam image dataset. With the key image filtering criteria, we generated the new spam image as follows.

- Improved the spam image to have the same texture as a ham image by adding a background layer and increase the entropy of the local binary pattern.
- The color elements of the spam image are made closer to the color histogram of the camera generated picture.
- Introduced some noise to the image as the natural image contains more noise compared to current spam.
- Finally, modified the image metadata to make it even with the compression ratio of genuine images.

We created 1029 such images and Figure 38 displays two such images from our dataset. This dataset is extensively tested with our two proposed techniques. We compare the


Not having Adsense Account? Maybe you can try this,
Chitika + Infolinks a Better Adsense Alternative
Below I mentioned some 15+ Best In-Text Advertising Programs with their minimum payment and payout details. Share & RT them if you like it.
Advertise on RainedOut Messages
& Reach New Customers.
In order to keep RainedOut a free
service for organizations
and its subscribers, we rely on
message sponsors to help cover
the cost of the service.

(a) (b)

Figure 38: New Spam Images

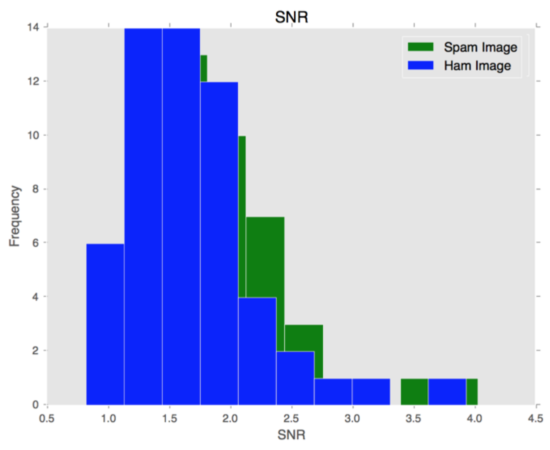
results with the two existing datasets.

6.6.1 SVM-based Detection

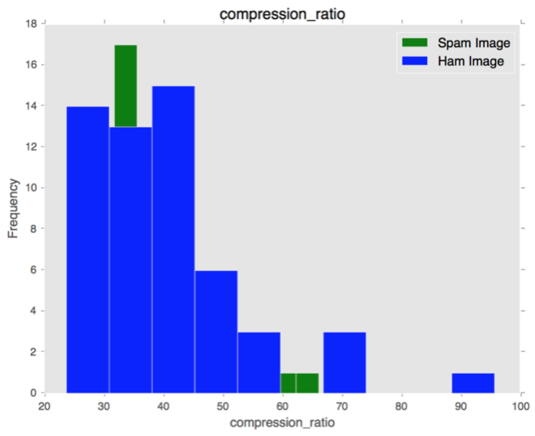
We consider a training set with 500 newly created spam images and 500 ham images. We use a total of 839 images from both classes for the test set. We first extracted the features of the images and plotted the frequency as in Figure 39. It is observed that there is no clear cut line between the spam and the ham. These feature distributions have been further analyzed by computing AUC and plotting them with a comparison to the AUC values obtained for dataset 1 as shown in Figure 40. It is observed the classification capabilities of the individual features have been reduced.

We classified these feature vector using all kernel functions of SVM and results are shown in Table 13. It is observed that a very low detection rate is achieved using this dataset when compared to the other two datasets.

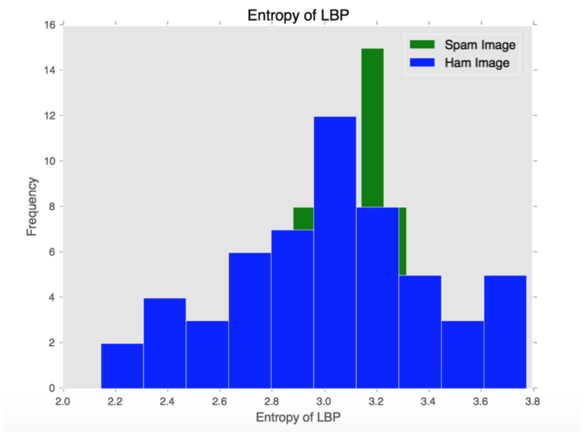
The AUC value obtained is 0.76 , and the ROC curve for the classification is



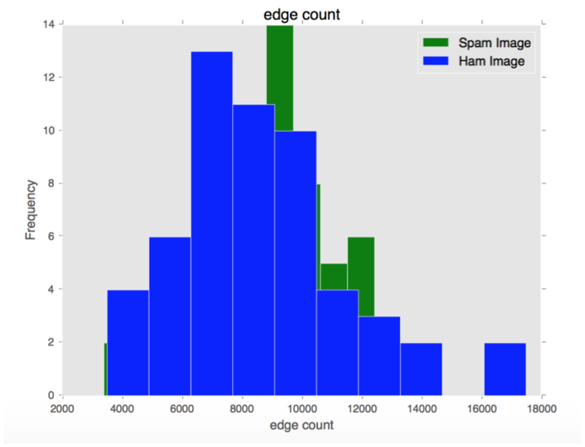
(a) Signal to noise ratio



(b) Compression ratio



(c) LBP



(d) Edge Count

Figure 39: Feature distribution for spam versus ham images for dataset 3

Table 13: Accuracy and FPR with different Kernel functions for dataset 3

Kernel	**Accuracy**	**FPR**
Linear	0.65	0.21
RBF	0.76	0.12
Polynomial	.72	0.19

presented in Figure 41.

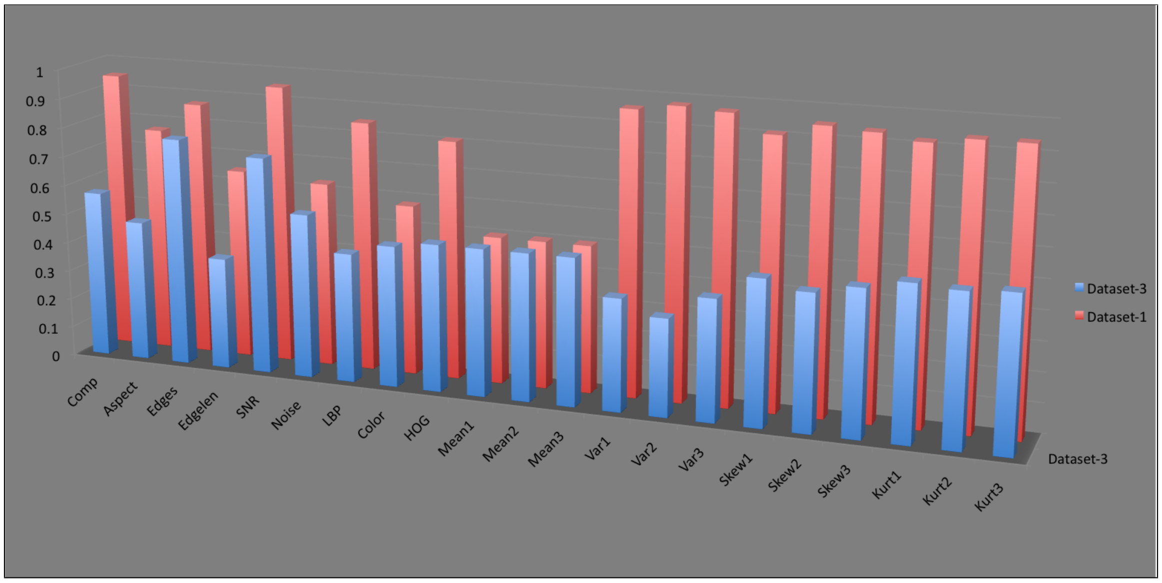


Figure 40: AUC for the feature distributions of dataset 1 and dataset 3

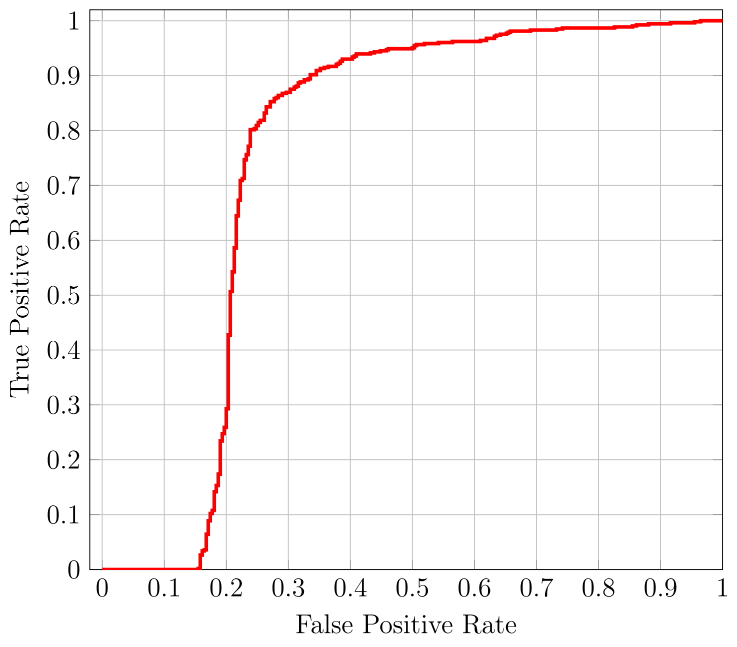


Figure 41: ROC Curve for SVM with AUC = 0.76 for dataset 3

6.6.2 PCA-based Detection

We consider 529 spam images as the train set and the test set with of 500 images from each class. Figure 42 displays the scatter plot for the Euclidean distance scores considering one Eigen component. Figure 43 shows the ROC curve and the corresponding AUC value of 0.48. Thus this method also produces a low detection rate for this dataset as compared to the existing datasets.

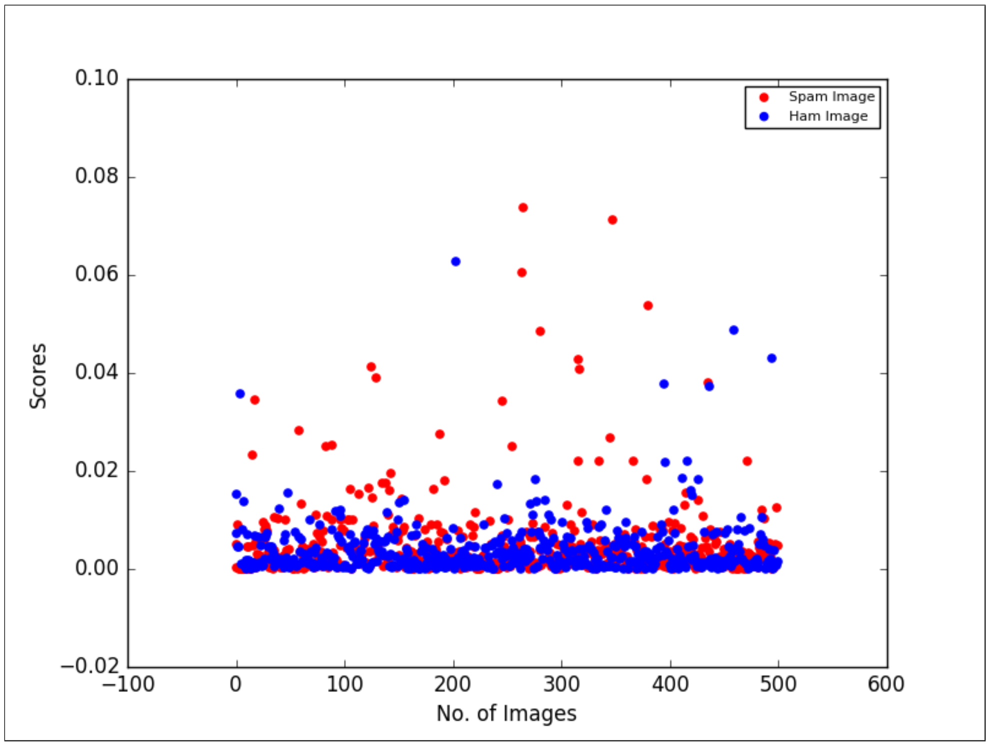


Figure 42: Scatterplot for Eigenspam Score for dataset 3

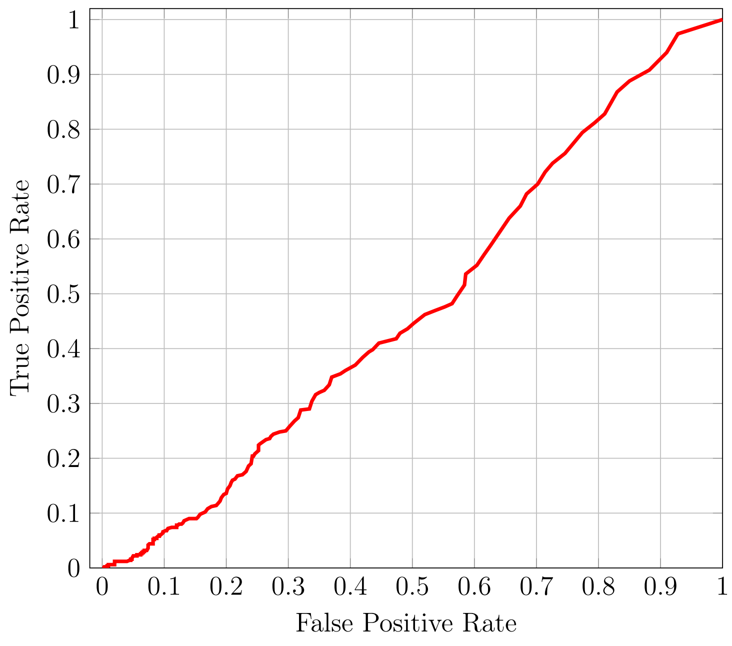
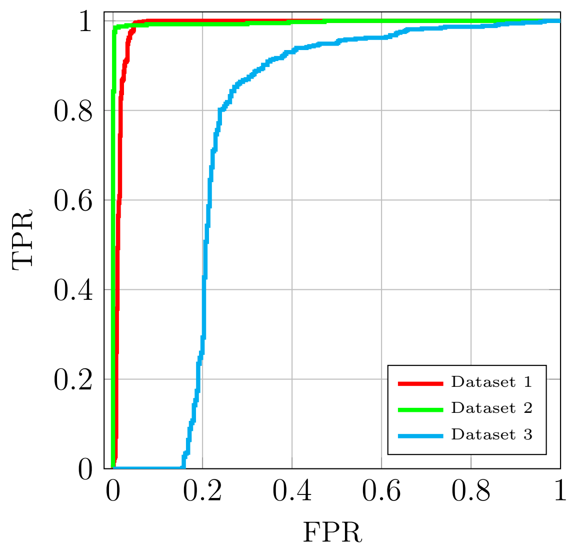


Figure 43: ROC Curve for Eigenspam with AUC = 0.48 for dataset 3

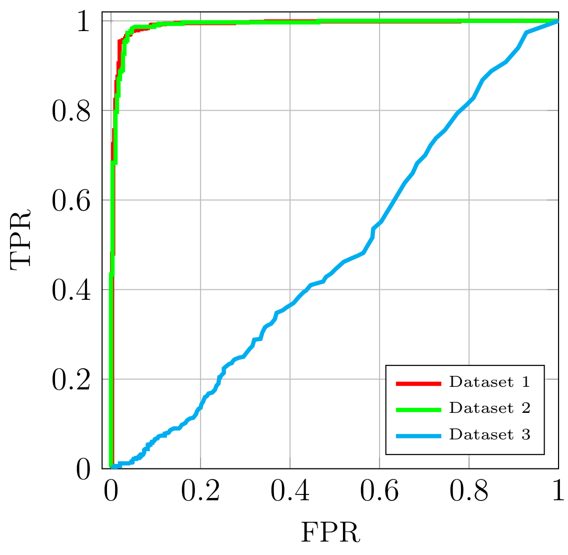
6.7 Cumulative Analysis

Finally, in this section we analyze the cumulative results of the proposed image spam detection techniques as shown in the Table 14 and Figure 44. From the SVM-based detection experiments, it is observed that using feature selection we achieve a better accuracy rate of 0.97 compared to the result without feature selection which is 0.96 for dataset 1. And the results obtained with the second dataset show a perfect classification rate with an AUC value of 1. It is thus evident that the model employs a sensible approach for feature selection based on the discriminant properties of the training set and reduces the effort. A broad feature set that includes all the aspects of the image would ideally yield a good result for this technique.

On the other hand, using PCA-based detection we obtained similar detection rates as the SVM technique with an AUC of 0.99 for dataset 1. But, Eigenspam



(a) SVM-based detection



(b) PCA-based detection

Figure 44: ROC curves for our proposed approaches for all datasets

Table 14: Cumulative Results

Datasets	**PCA**	**SVM**		
	AUC	**AUC**	**Accuracy**	**FPR**
Dataset 1	0.99	0.99	0.97	0.04
Dataset 2	0.99	1.00	0.99	0.01
Dataset 3	0.48	0.76	0.76	0.21

has a comparatively low computational complexity, though SVM model reduces the effort using feature selection. Though the preprocessing step requires a minimal effort of resizing the image and converting it to greyscale, this model does not depend on the extraction of various image features but instead processes the image as a whole. Thus by reducing the overhead of identifying and selecting various image features, PCA-based detection outperforms the previous technique.

On the contrary to an efficient detection, the experiments with our new dataset have shown a decline in the productive filtering which is expected. As the spam set was constructed with an aim to circumvent the existing filters and in future would help design a stronger filter. With the feature distributions for the new set, it is apparent that the most discriminant feature values have been diminished to achieve this feat.

Furthermore, in comparison with the previous works related to image spam detection, our approaches have obtained a greater rate of detection and a minimal false positive rate. Considering dataset 1, we achieved a maximum accuracy of 0.97 and FPR of 0.04 which implies that our proposed systems are highly efficient with a low computational effort compared to the existing work.

CHAPTER 7

Conclusion and Future Work

The ever increasing use of electronic communication always poses a threat to the cyber world and demands a robust email spam detection system. Although there have been a numerous attempts in the past to detect an image spam, dynamic detection is always a challenge. In this project, we presented two novel approaches for efficient image spam filtering which employs active machine learning techniques.

Support Vector Machines have proven to be the best among the highly discriminative supervised learning algorithms. Hence, we based our first model on SVM technique that uses an active feature selection algorithm by manipulating its weight vectors. For this approach, we first extract a broad set of 21 discriminative image features relying on the fact that spam image possesses different characteristics compared to a genuine image. Next, we select the most influencing features using Linear SVM weights and generate a model. We classify images by extracting only the selected subset of features that decreases the computational effort. We further used the optimal feature set to develop a challenging class of image spam. Our second approach based on PCA, called the Eigenspam technique, is motivated by Eigenfaces which is widely used for facial recognition. This system generates eigenvectors, projects them onto the eigenspace and computes the Euclidean distance as a scoring factor to classify images.

Extensive experimental evaluations of both the methods on different datasets demonstrated the efficacy of the proposed system. From the results, it is observed that the SVM-based model employs a sensible approach for feature selection by actively learning the properties of the images and achieved a greater accuracy with a low false positive rate compared to the previous work. Although this system provides a fast classification, the intuitive solution based on PCA illustrates a reasonably low computational effort by reducing the complexity of feature extraction from the images and with the same accuracy rate. And with the newly generated dataset, SVM-based detection has performed reasonably better that the PCA-based approach.

Our future works may include, but not necessarily limited to combining our system with text localization to further identify and reduce the false positive rate. One such detection could be using the edges to isolate the words and filter the undesirable text. Also, new and efficient feature selection algorithms can be explored like using Principal Component Analysis for dimensionality reduction. Another potential approach could be classification based on clustering technique and combining it with a genetic algorithm to improve the seed value. Furthermore, investigating more discriminative features for spam images using our newly created dataset, which could be a valuable tool in improving the image spam detection capabilities. Since image spam is also widespread in different platforms such as mobile and social networks, our proposed system could be extended to detect such spam images. Finally, there is always room for an image spam corpus for extensive research.

LIST OF REFERENCES

[1] A. P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, *Pattern recognition*, 30(7):1145–1159, 1997

[2] J. Brownlee, An Introduction to Feature Selection, October 6, 2014 `http://machinelearningmastery.com/an-introduction-to-feature-selection`, Accessed on April 11, 2016

[3] J. Canny, A computational approach to edge detection, *Pattern Analysis and Machine Intelligence, IEEE Transactions*, (6):679–698, 1986

[4] C. C. Chang and C. J. Lin, LIBSVM: a library for support vector machines, *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011

[5] O. Chapelle, V. Vapnik, O. Bousquet and S. Mukherjee, Choosing multiple parameters for support vector machines, *Machine learning*, 46(1-3):131-59, 2002

[6] S. Deshpande, Y. Park, and M. Stamp, Eigenvalue analysis for metamorphic detection, *Journal of Computer Virology and Hacking Techniques*, 10(1):53–65, 2014

[7] M. Dredze, R. Gevaryahu and A.Elias-Bachrach, Learning Fast Classifiers for Image Spam, *CEAS*, 2007

[8] M. Dredze, Image spam dataset, 2007, `http://www.cs.jhu.edu/~mdredze/datasets/image_spam/`, Accessed on February 15, 2016

[9] Y. Gao, A. Choudhary and G. Hua, A comprehensive server to client side approach to image spam detection, *IEEE Transactions on Information Forensics and Security*, 5(4), 2010

[10] Y. Gao and A. Choudhary, Active learning image spam hunter, *Advances in Visual Computing*, 293-302, 2009

[11] Y. Gao, M. Yang and A. Choudhary, Semi supervised image spam hunter: A regularized discriminant em approach, *Advanced Data Mining and Applications*, 152–164, 2009

[12] Y. Gao, M. Yang, X. Zhao, B. Pardo, Y. Wu,T. N. Pappas and A. Choudhary, Image spam hunter, *Acoustics, Speech and Signal Processing*, 2008, *ICASSP*, 2008, *IEEE International Conference*, 1765–1768, 2008

[13] Y. Gao, Image spam hunter dataset, 2008
http://www.cs.northwestern.edu/~yga751/ML/ISH.htm, Accessed on September 20, 2015

[14] I. Guyon, J. Weston, S. Barnhill and V. Vapnik, Gene selection for cancer classification using support vector machines, *Machine learning*, 46(1-3):389–422, 2002

[15] M. A. Hearst, S. T. Dumais, E. Osman, J. Platt, and B. Scholkopf, Support vector machines, *Intelligent Systems and their Applications, IEEE*, 13(4):18–28, 1998

[16] P. He, X. Wen, and W. Zheng, A simple method for filtering image spam, *Computer and Information Science, ICIS,Eighth IEEE/ACIS International Conference*, 910–913), 2009

[17] History of Email,
http://www.inventorofemail.com/history_of_email.asp, Accessed on April 25,2016

[18] IBM X-Force Threat Intelligence Report 2016
http://www-03.ibm.com/security/xforce/, Accessed on April 15, 2016

[19] U. Jain and S. Dhavale, Image Spam Detection Technique based on Fuzzy Inference system, Masters Report, Department of Computer Engineering, Defense Institute of Advanced Technology, 2015

[20] Kaspersky Securelist - Information about viruses, threats and spam,
https://securelist.com/threats/spam-and-phishing/, Accessed on March 10, 2016

[21] S. Kullback and R.A. Leibler, On information and sufficiency, *The annals of mathematical statistics*, 22(1):79–86, 1951

[22] T. Mäenpää and M. Pietikäinen, Texture analysis with local binary patterns, *Handbook of Pattern Recognition and Computer Vision*, 3:197–216, 2005

[23] Mean vector and covariance matrix, NIST,
http://www.itl.nist.gov/div898/handbook/pmc/section5/pmc541.htm, Accessed on January 20, 2016

[24] D. Mladeni, J. Brank, M. Grobelnik andN. Milic-Frayling, Feature selection using linear classifier weights: interaction with classification models, *Proceedings of the 27th annual international ACM SIGIR conference on Research and Development in Information Retrieval*, 234–241, 2004

[25] T. T. Ng, S.F. Chang, and M.P. Tsui, Lessons learned from online classification of photo-realistic computer graphics and photographs, *IEEE Workshop on Signal Processing Applications for Public Security and Forensics (SAFE)*, 2007

[26] M. Nixon, *Feature extraction & image processing*, Academic Press, 2008

[27] W.H. Press, *Numerical recipes: The art of scientific computing*, Third edition, Cambridge university press, 2007

[28] Princeton Spam Image Benchmark, `http://www.cs.princeton.edu/cass/spam/`, Accessed on May 3, 2016

[29] A. Rakotomamonjy, Variable selection using SVM based criteria, *The Journal of Machine Learning Research*, 3:1357–70, 2003

[30] Scikit-Learn: Machine learning in python. `http://scikit-learn.org/stable/` Accessed on October 3, 2015

[31] J. Shlens, A Tutorial on Principal Component Analysis, `http://www.cs.cmu.edu/~elaw/papers/pca.pdf`, Accessed on March 5, 2016

[32] Spam Assassin `http://spamassassin.apache.org/` Accessed on March 1, 2016

[33] Spam SMS, `http://www.smh.com.au/technology/security/attack-of-the-spasms-the-new-sms-danger-20090630-d369.html`, Accessed on April 5, 2016

[34] Spam Statistics, `https://www3.trustwave.com/support/labs/spam_statistics.asp`, Accessed on April 15, 2016

[35] M. Stamp, *Machine Learning with Applications in Information Security*, first edition, unpublished manuscript, 2015

[36] Symantec trend report, `https://www.symantec.com/security_response/publications/monthlythreatreport.jsp#Spam`, Accessed on April 15, 2016

[37] M. Turk, A. Pentland, Eigenfaces for recognition, *Journal of cognitive neuroscience*, 3(1):71–86, 1991

[38] Z. Wang, W. K. Josephson, Q. Lv, M. Charikar and K. Li, Filtering Image Spam with Near-Duplicate Detection, *CEAS*, 2007

[39] Z. M. Win and N. Aye, Detecting Image Spam Based on File Properties, Histogram and Hough Transform, *Journal of Advances in Computer Networks*, 2(4), 2014

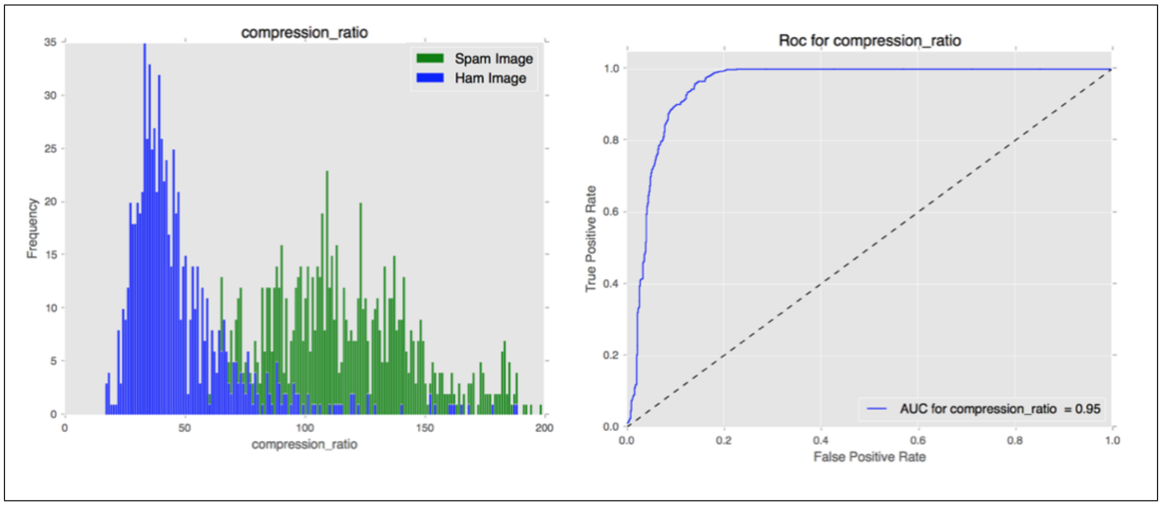
[40] Wolfram Mathworld, Covariance Matrix, `http://mathworld.wolfram.com/CovarianceMatrix.html`, Accessed on March 20, 2016

[41] Wolfram Mathworld, Vector Space Projection, `http://mathworld.wolfram.com/VectorSpaceProjection.html`, Accessed on March 1, 2016

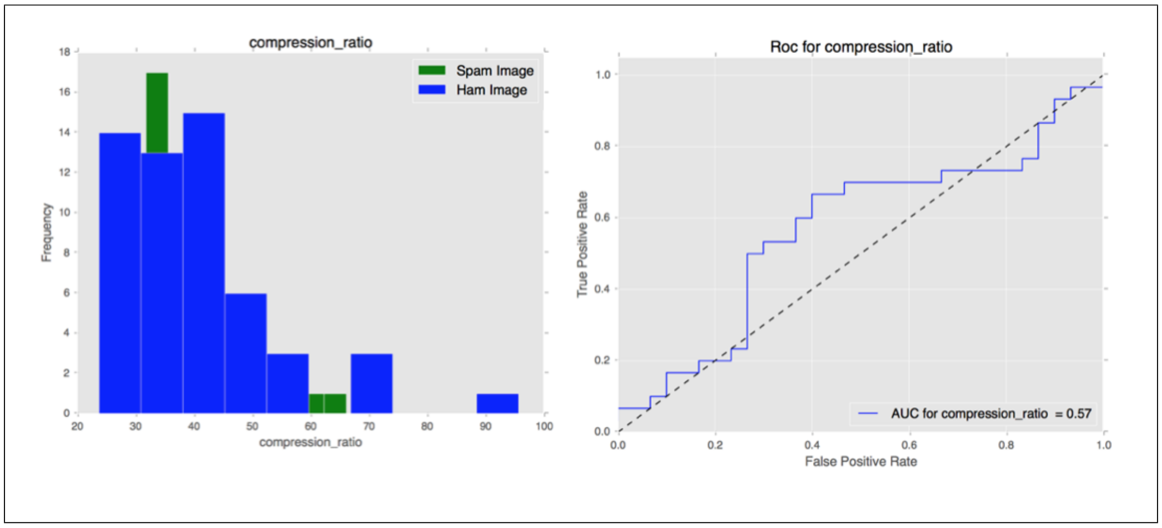
[42] Yale Face Database, `http://www.cad.zju.edu.cn/home/dengcai/Data/FaceData.html`, Accessed on March 5, 2016

APPENDIX A

Feature Distributions

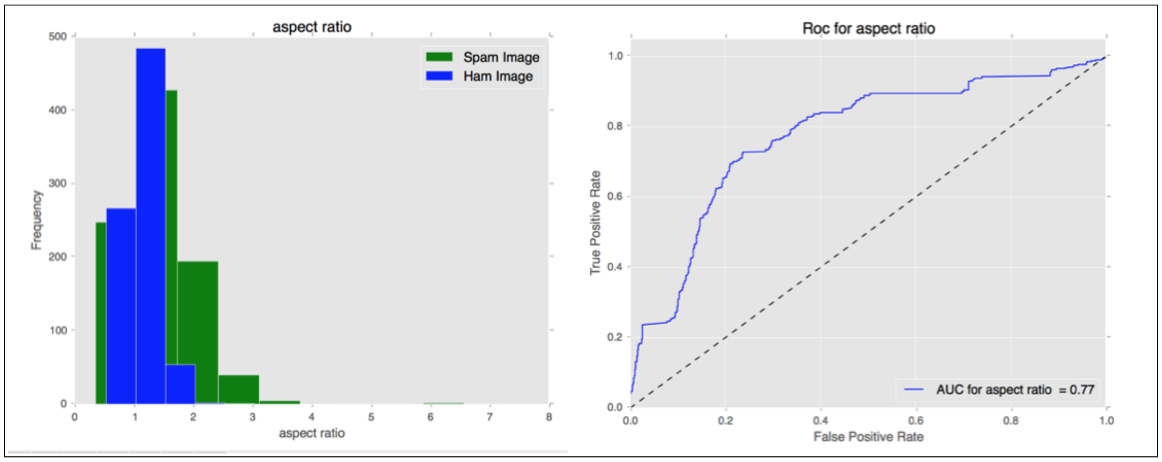


(a) Dataset 1

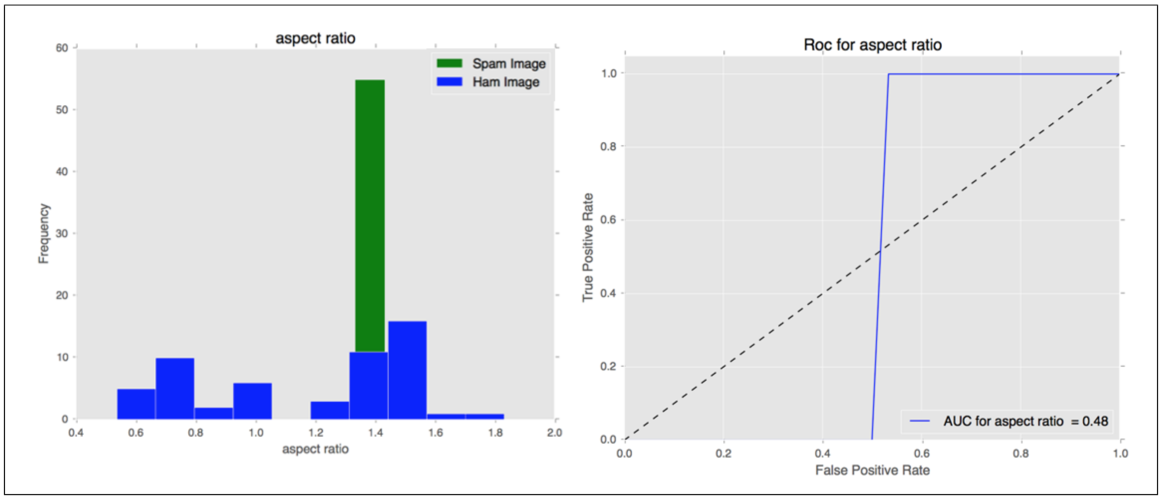


(b) Dataset 3

Figure A.45: Feature Distribution and ROC for Compression Ratio

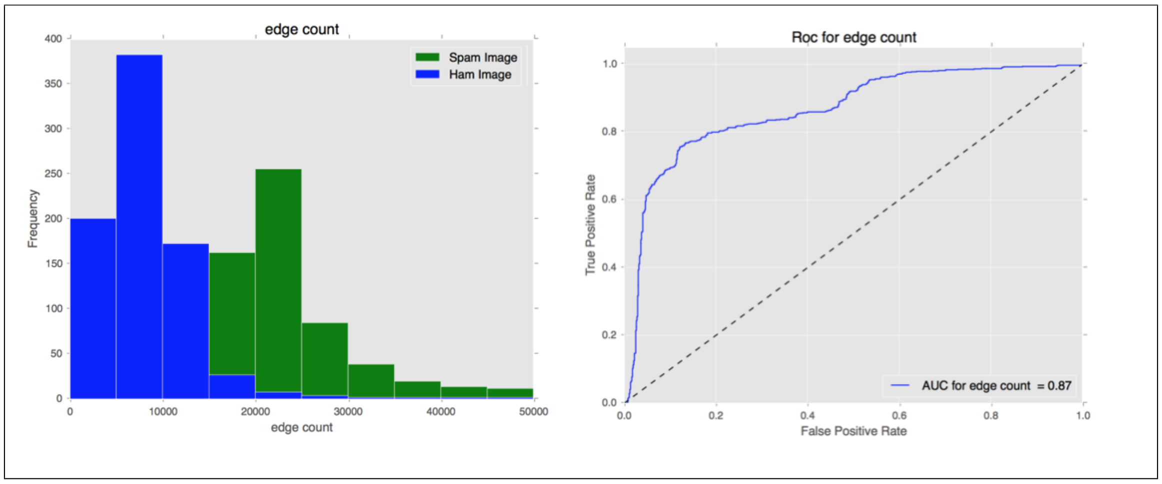


(a) Dataset 1

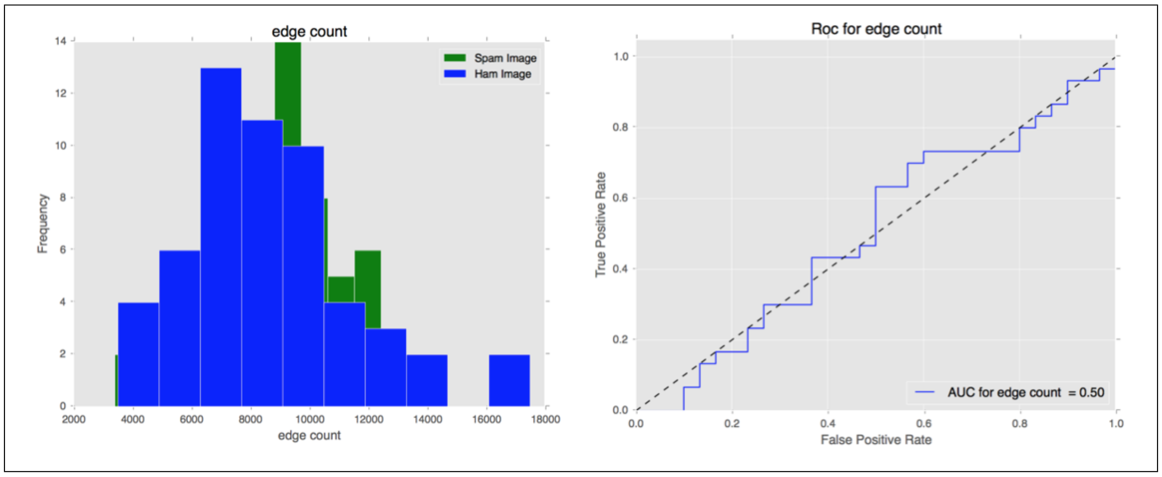


(b) Dataset 3

Figure A.46: Feature Distribution and ROC for Aspect Ratio

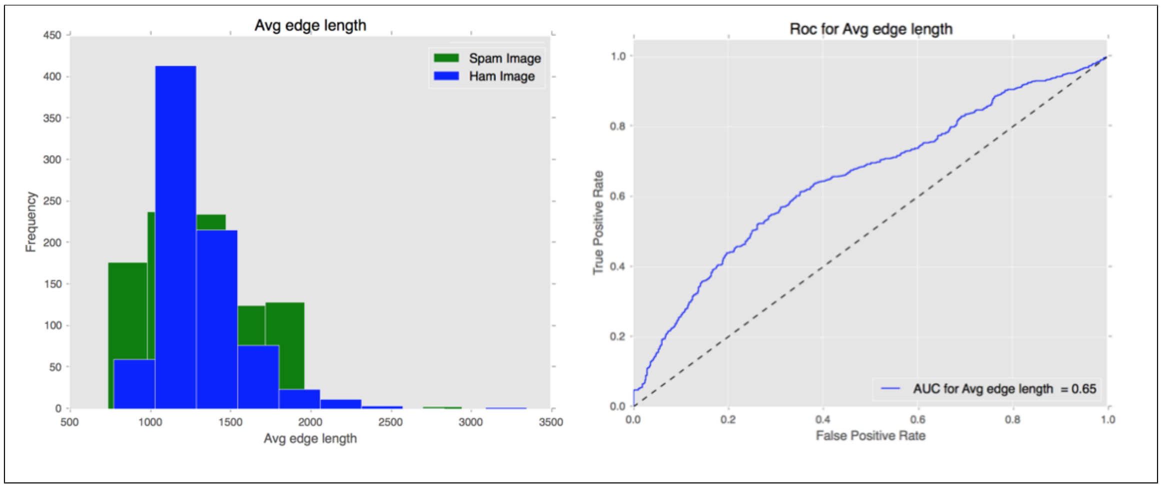


(a) Dataset 1

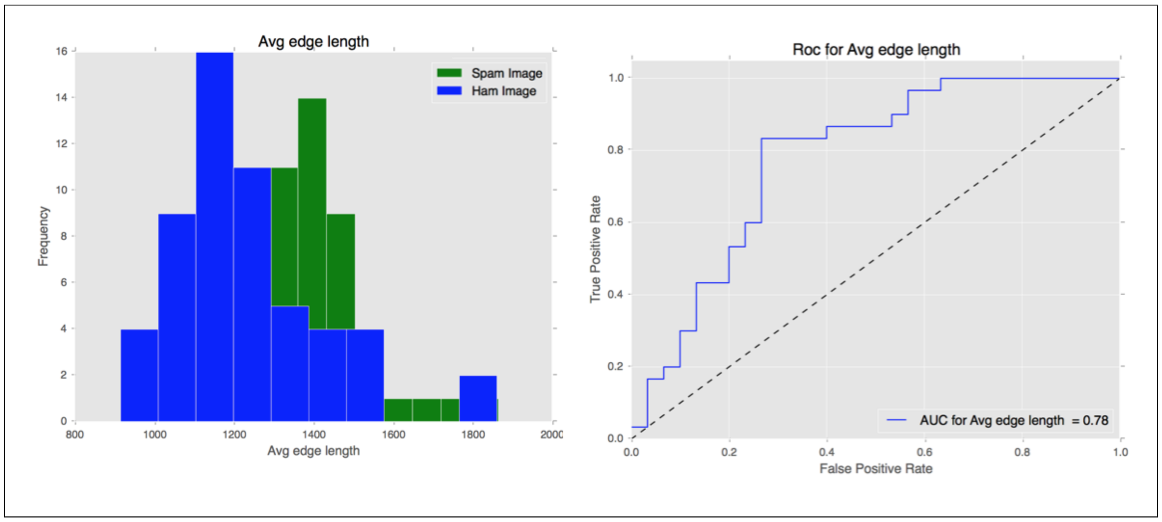


(b) Dataset 3

Figure A.47: Feature Distribution and ROC for Edge Count

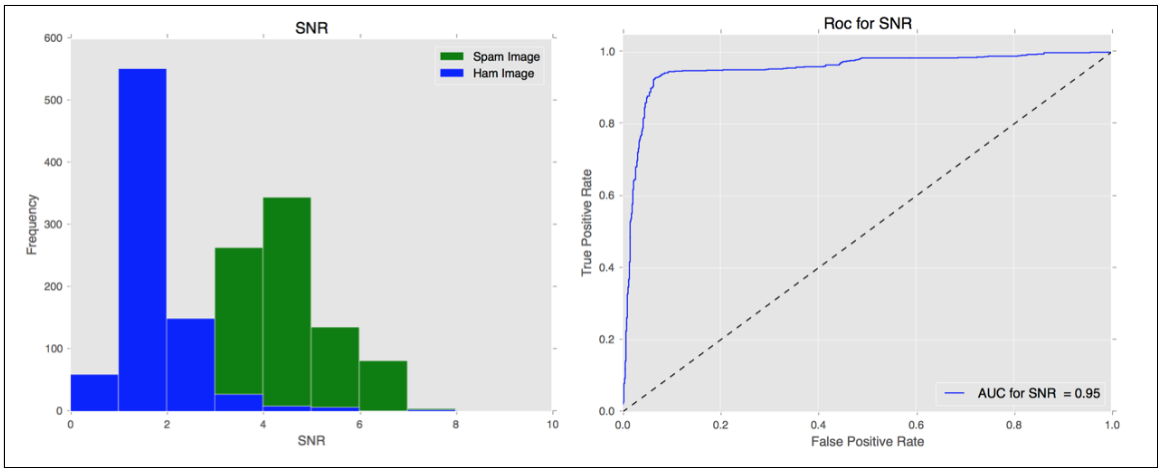


(a) Dataset 1

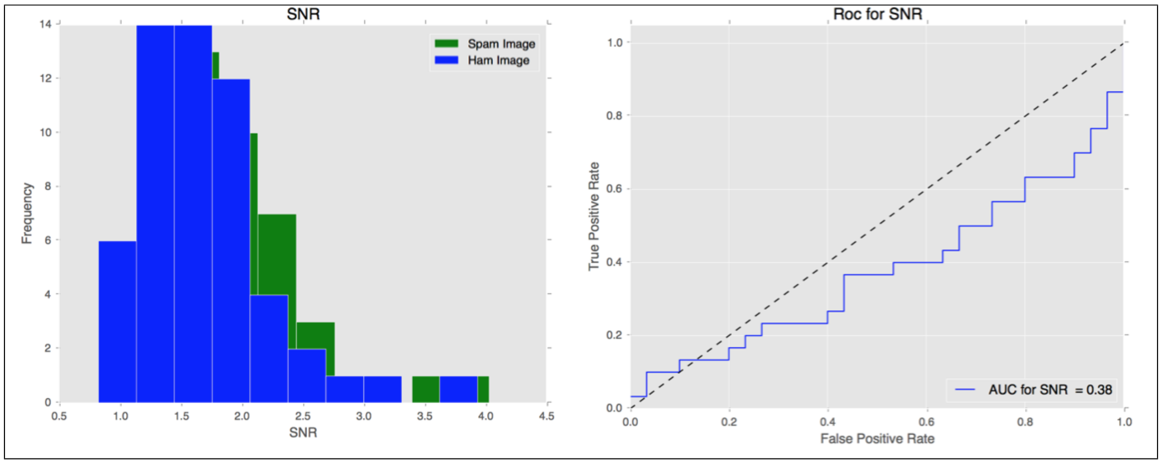


(b) Dataset 3

Figure A.48: Feature Distribution and ROC for Average Edge Length

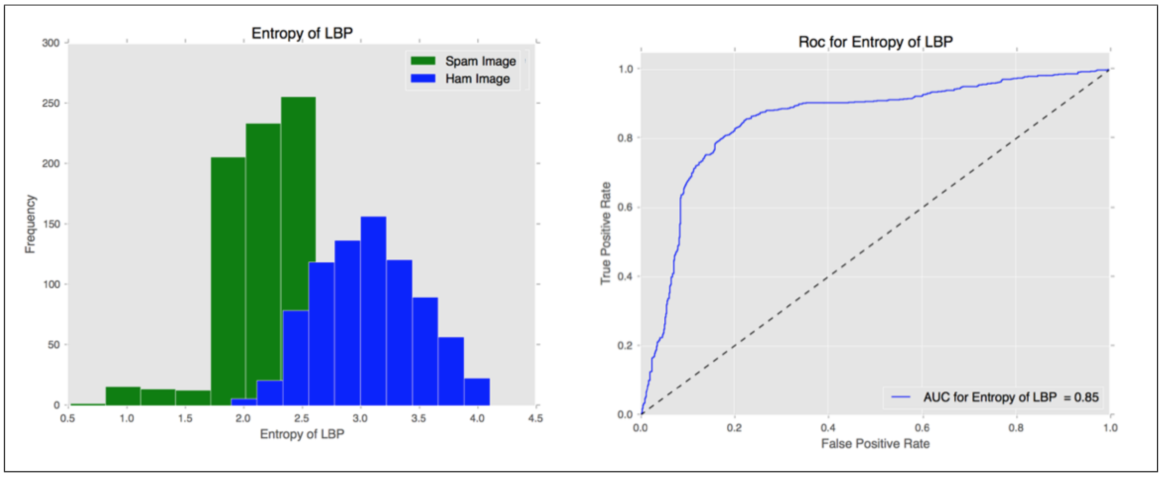


(a) Dataset 1

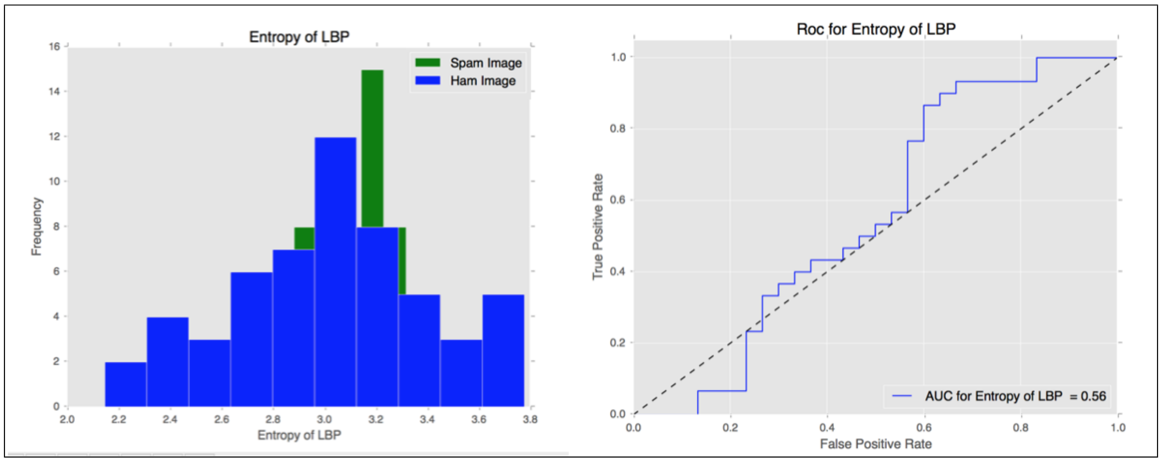


(b) Dataset 3

Figure A.49: Feature Distribution and ROC for Signal to Noise Ratio

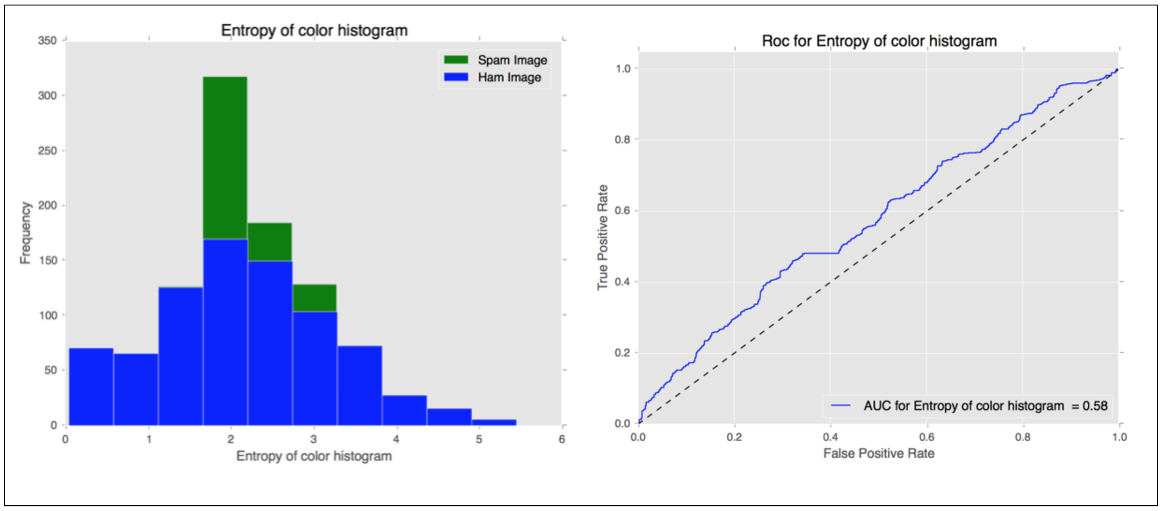


(a) Dataset 1

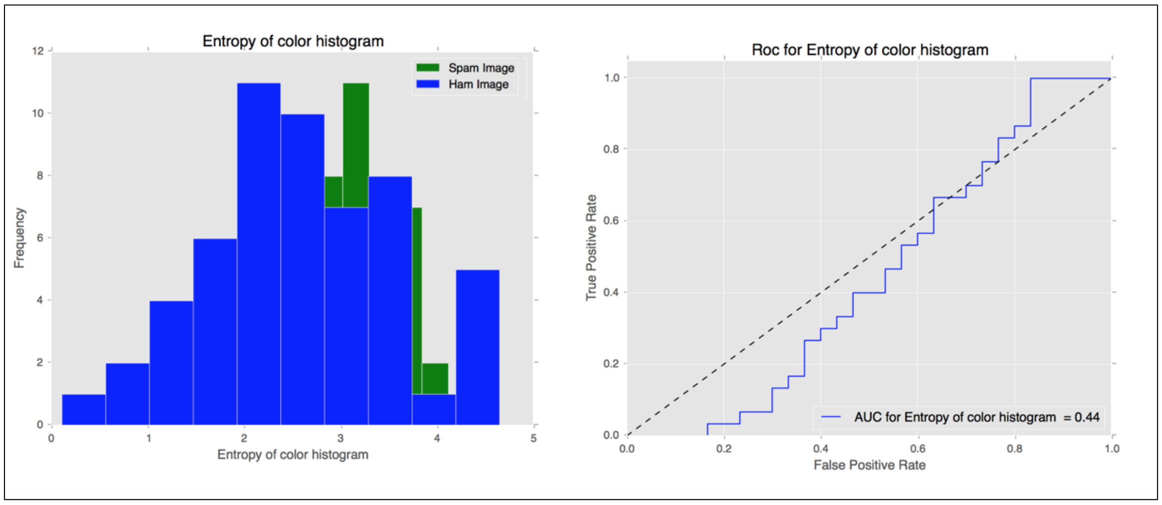


(b) Dataset 3

Figure A.50: Feature Distribution and ROC for Entropy of Local Binary Pattern

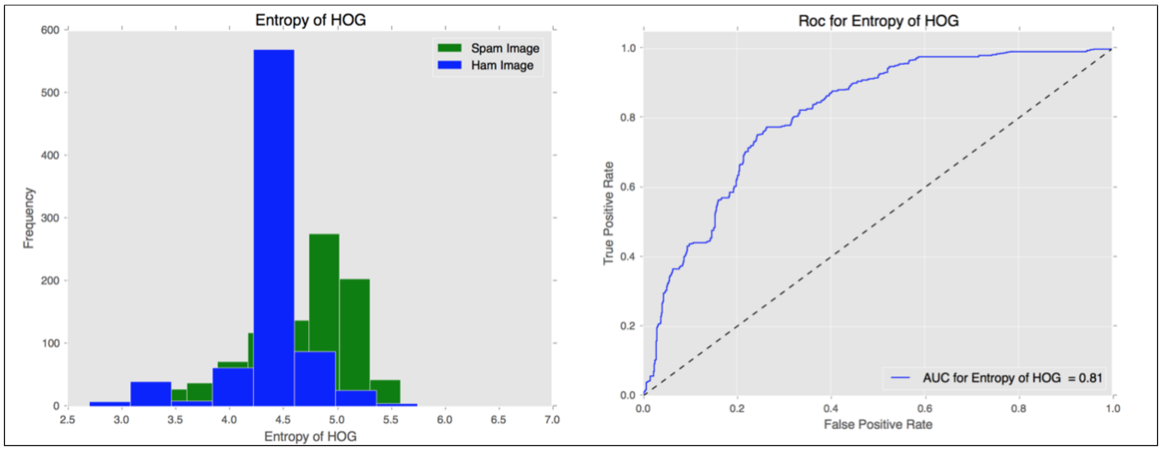


(a) Dataset 1

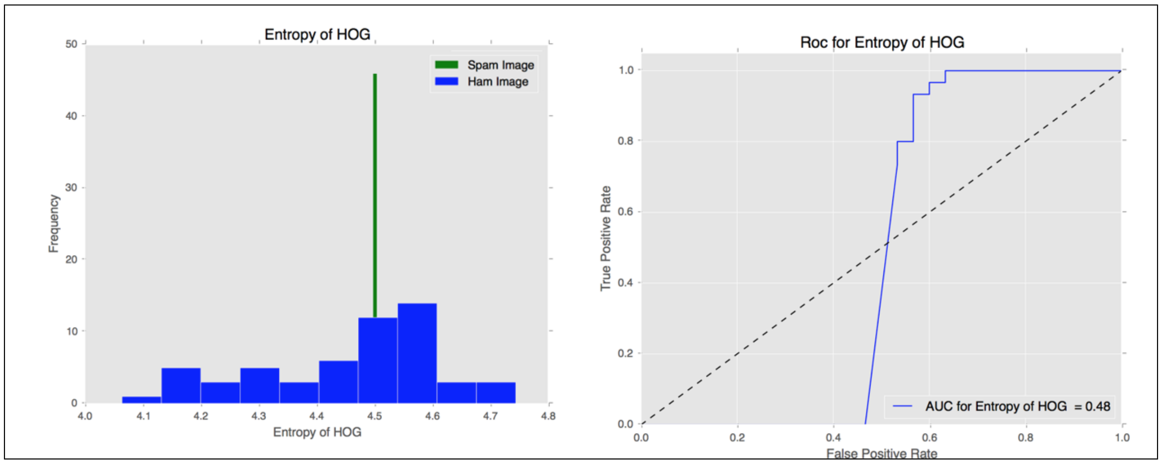


(b) Dataset 3

Figure A.51: Feature Distribution and ROC for Entropy of color histogram

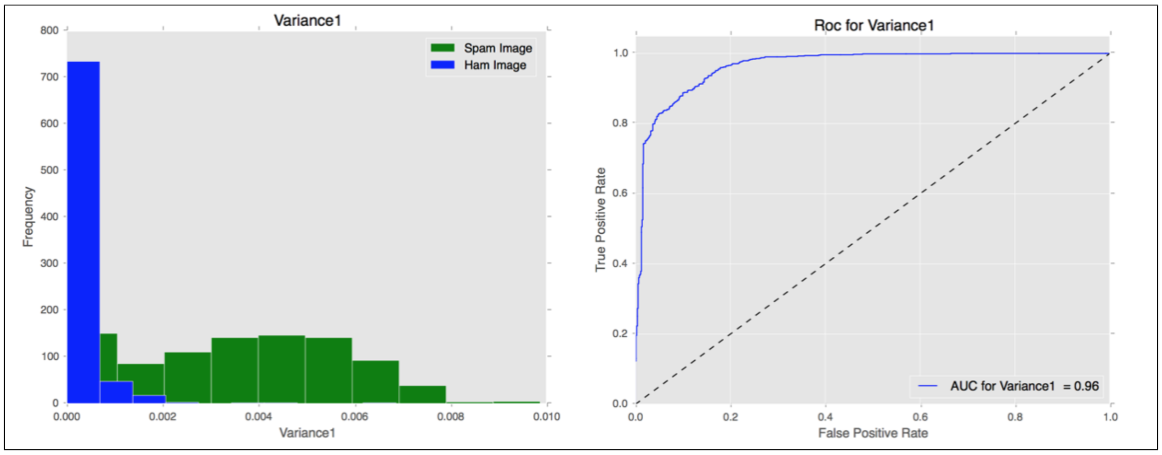


(a) Dataset 1

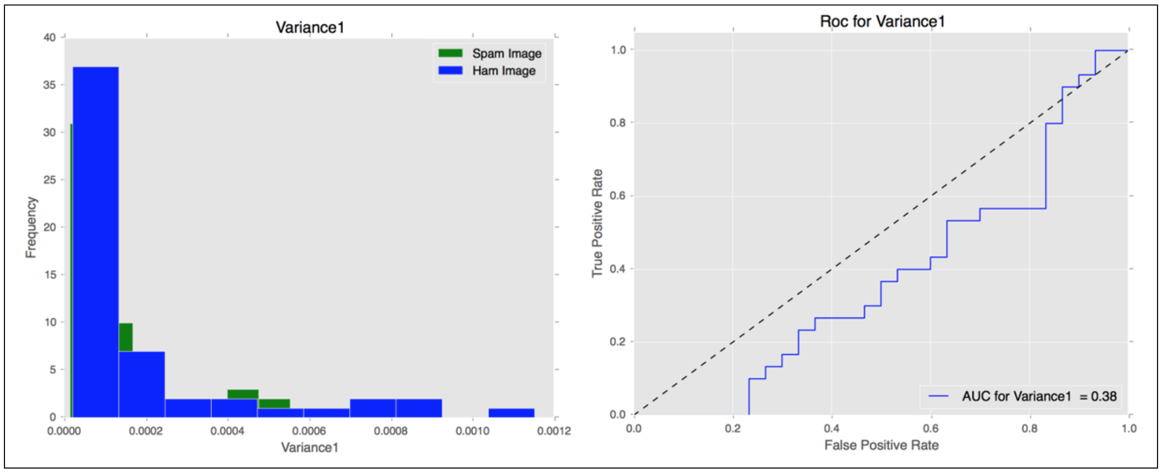


(b) Dataset 3

Figure A.52: Feature Distribution and ROC for Entropy of HOG

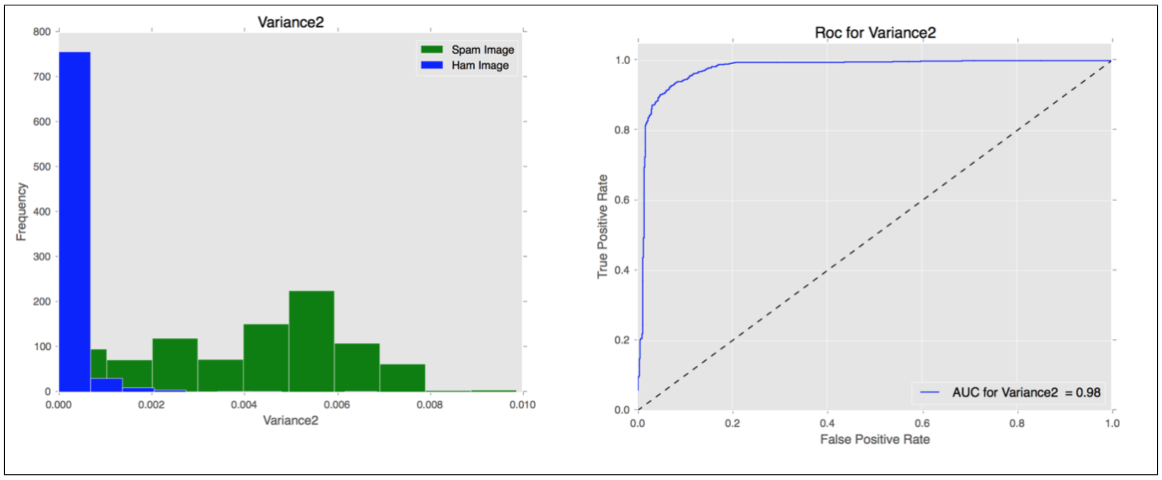


(a) Dataset 1

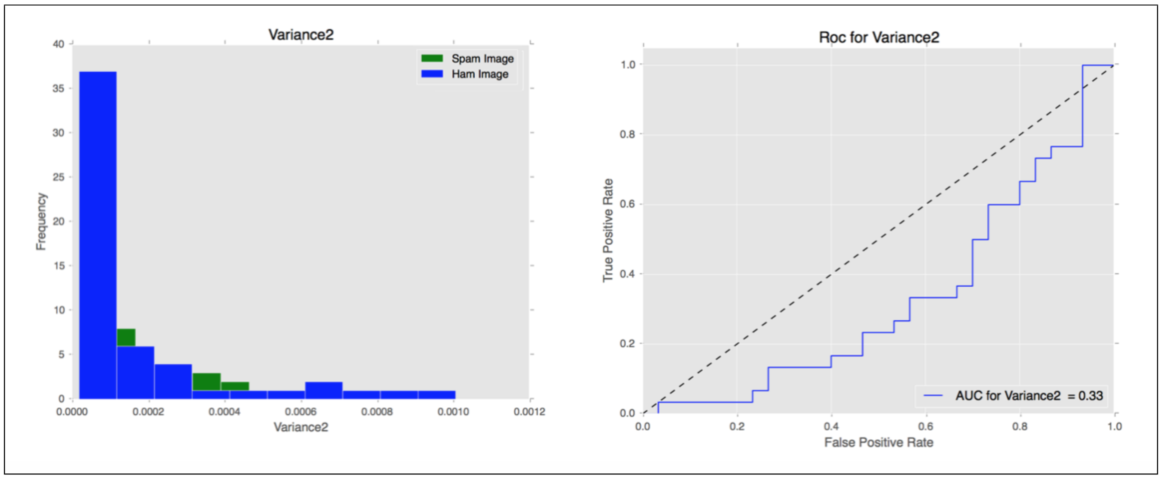


(b) Dataset 3

Figure A.53: Feature Distribution and ROC for Variance of First Color Channel

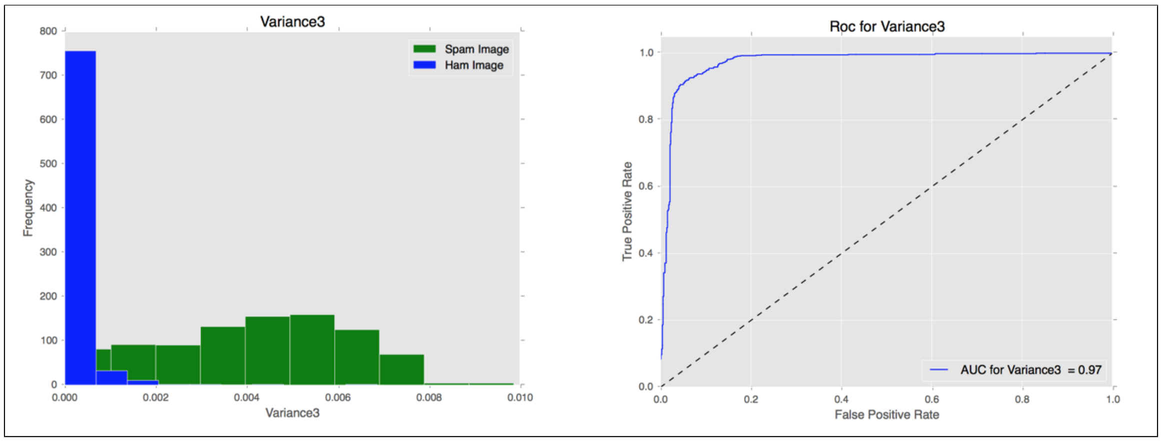


(a) Dataset 1

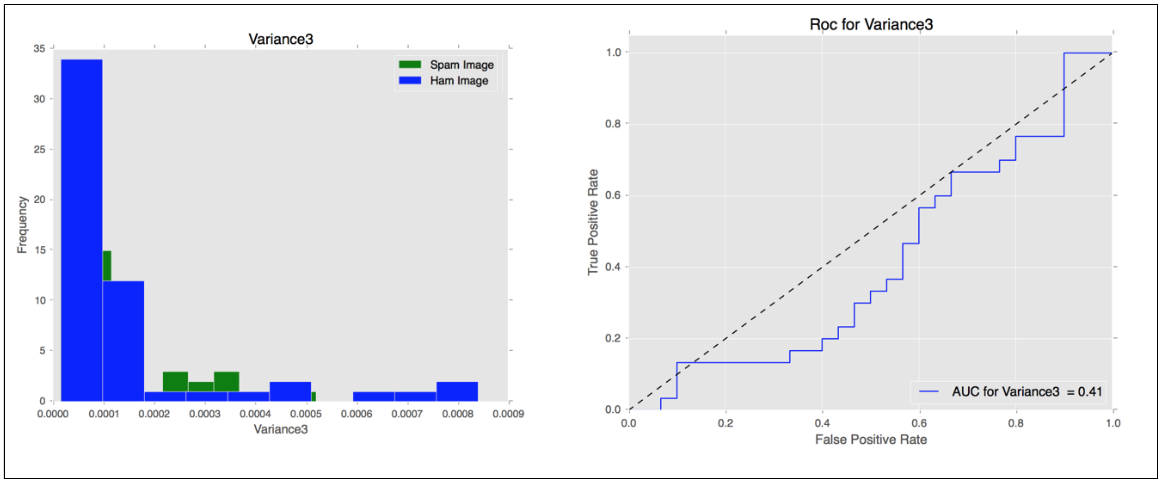


(b) Dataset 3

Figure A.54: Feature Distribution and ROC for Variance of Second Color Channel

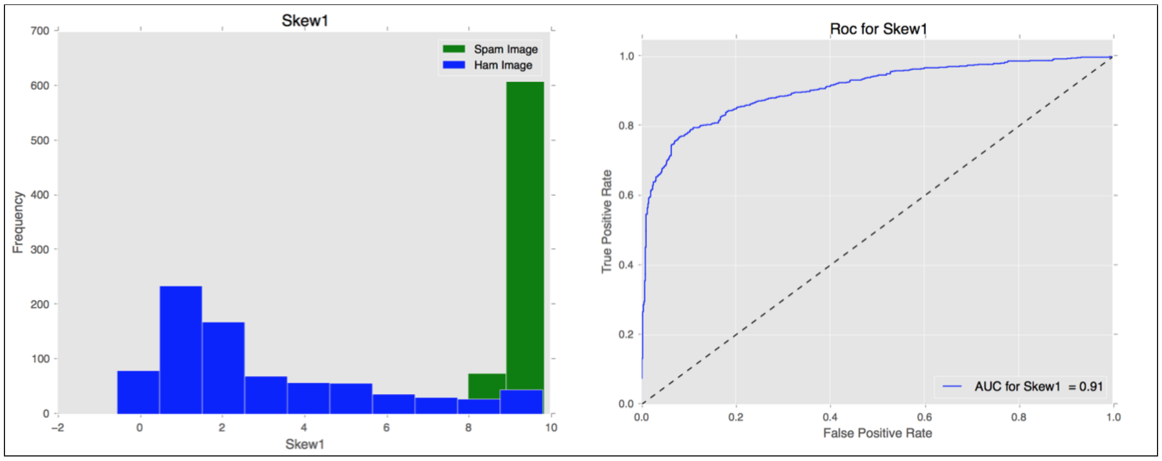


(a) Dataset 1

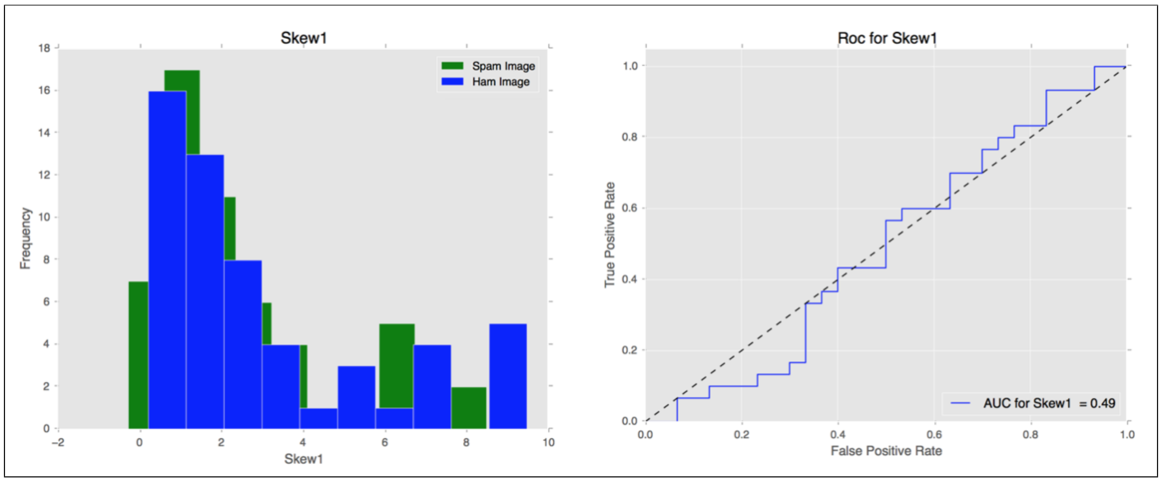


(b) Dataset 3

Figure A.55: Feature Distribution and ROC for Variance of Third Color Channel

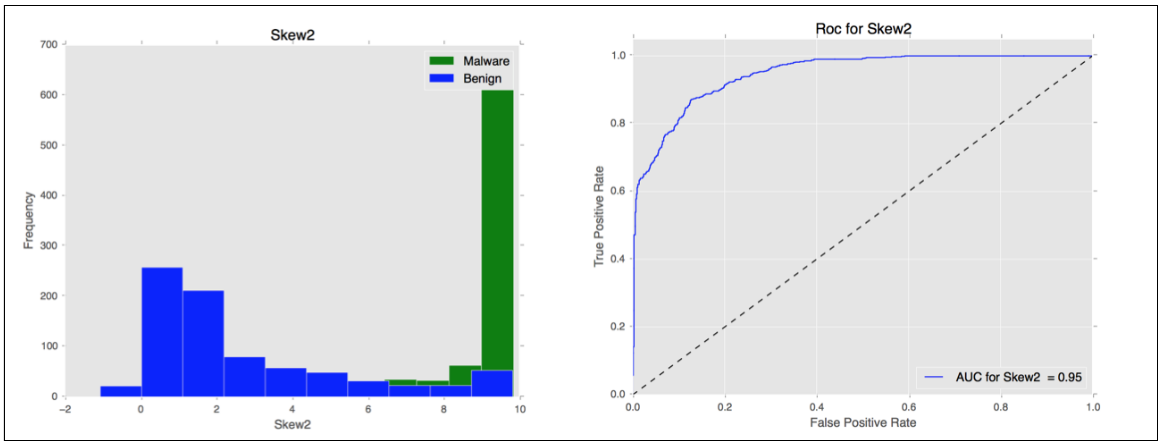


(a) Dataset 1

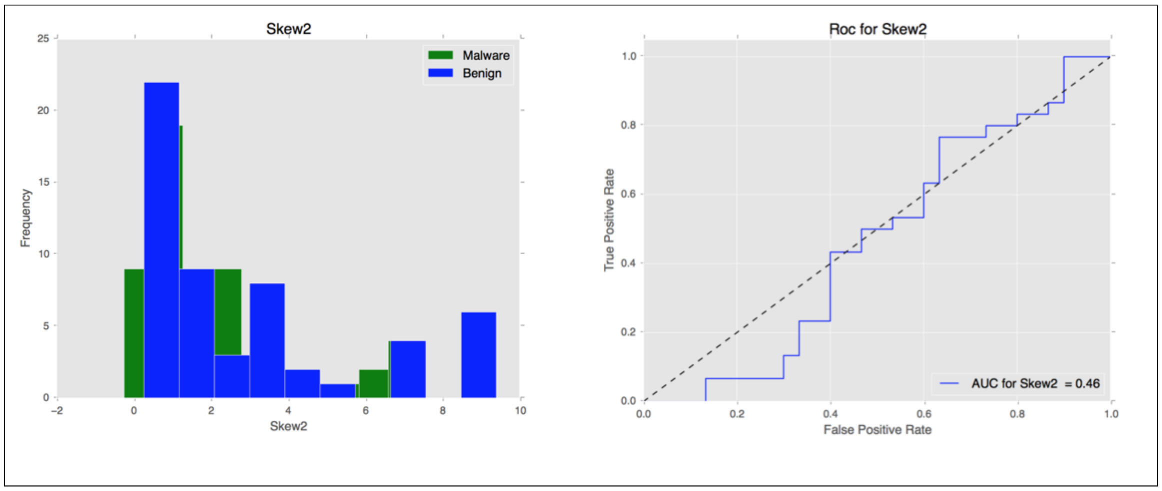


(b) Dataset 3

Figure A.56: Feature Distribution and ROC for Skew of First Color Channel

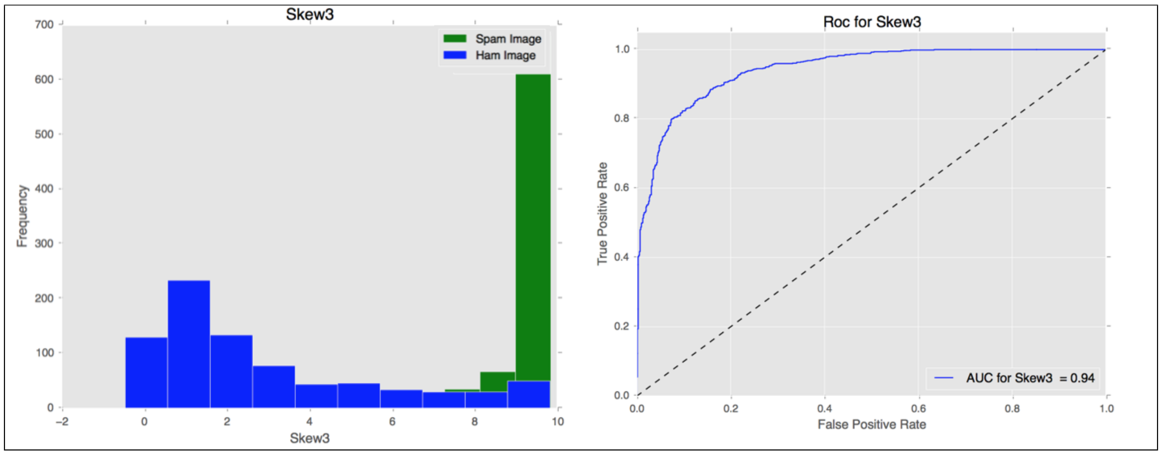


(a) Dataset 1

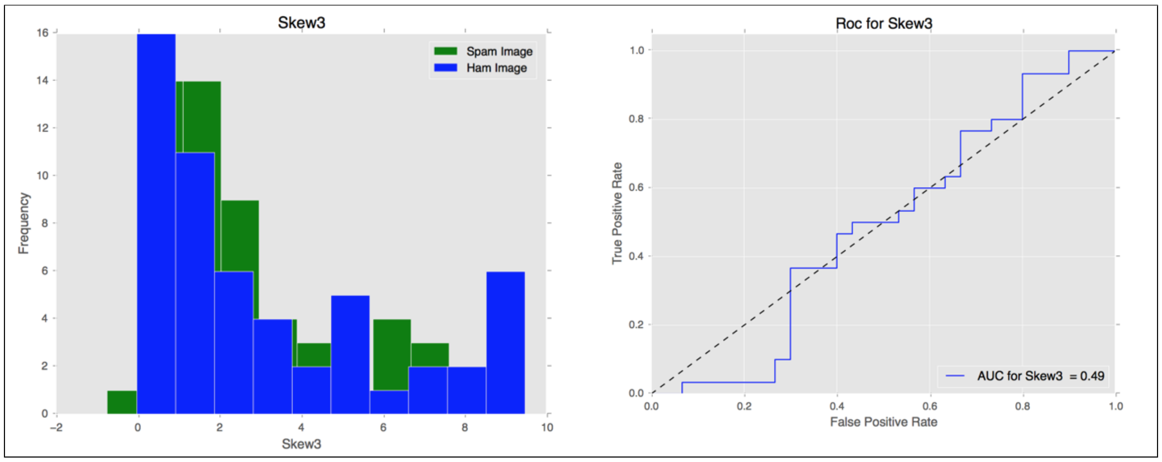


(b) Dataset 3

Figure A.57: Feature Distribution and ROC for Skew of Second Color Channel

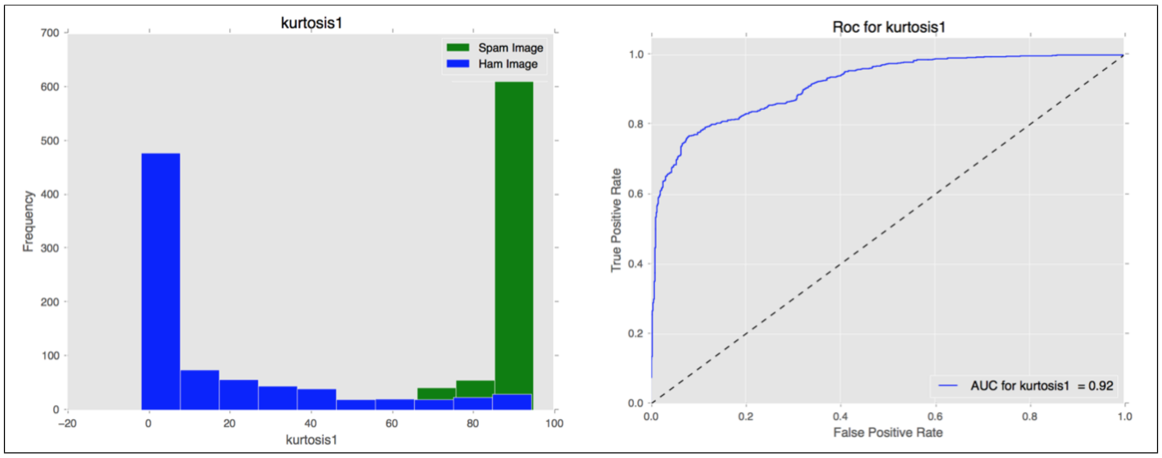


(a) Dataset 1

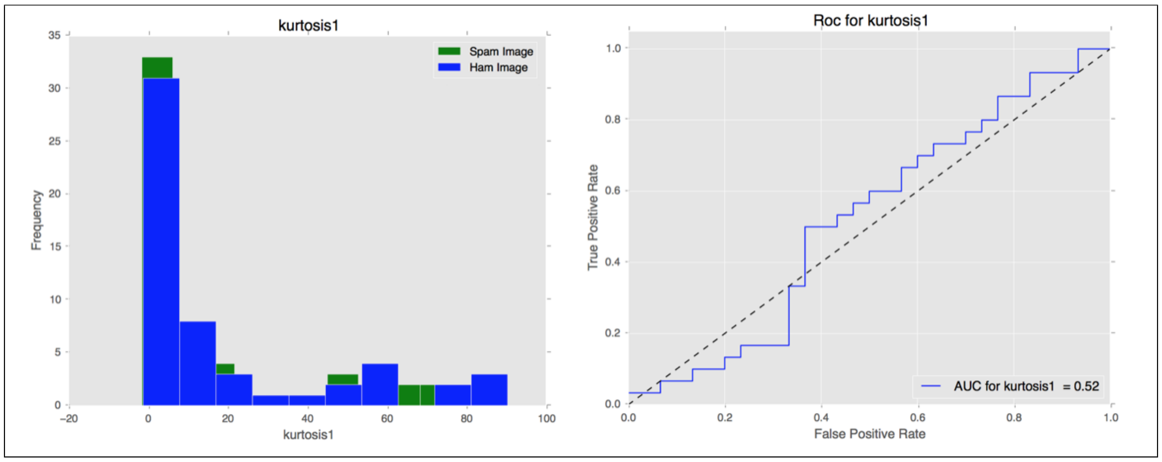


(b) Dataset 3

Figure A.58: Feature Distribution and ROC for Skew of Third Color Channel

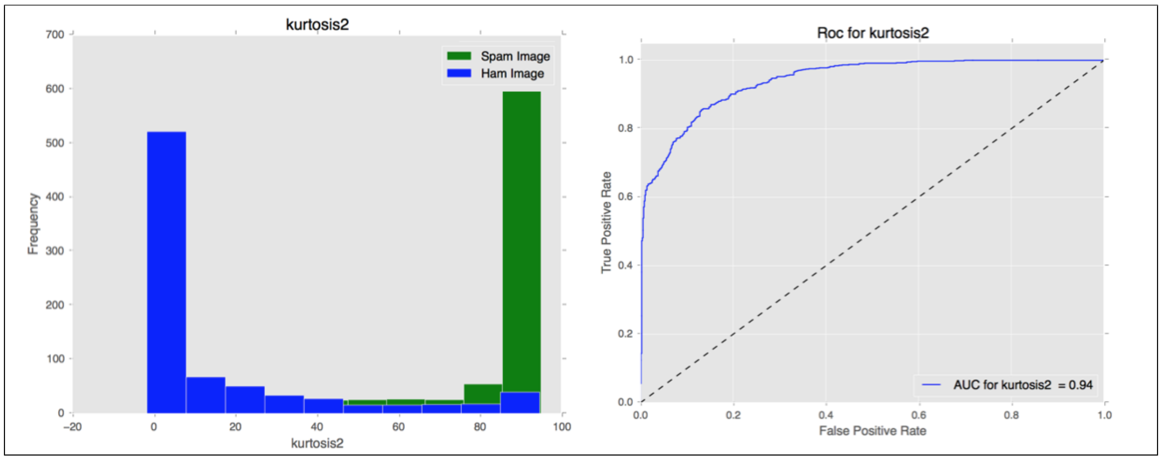


(a) Dataset 1

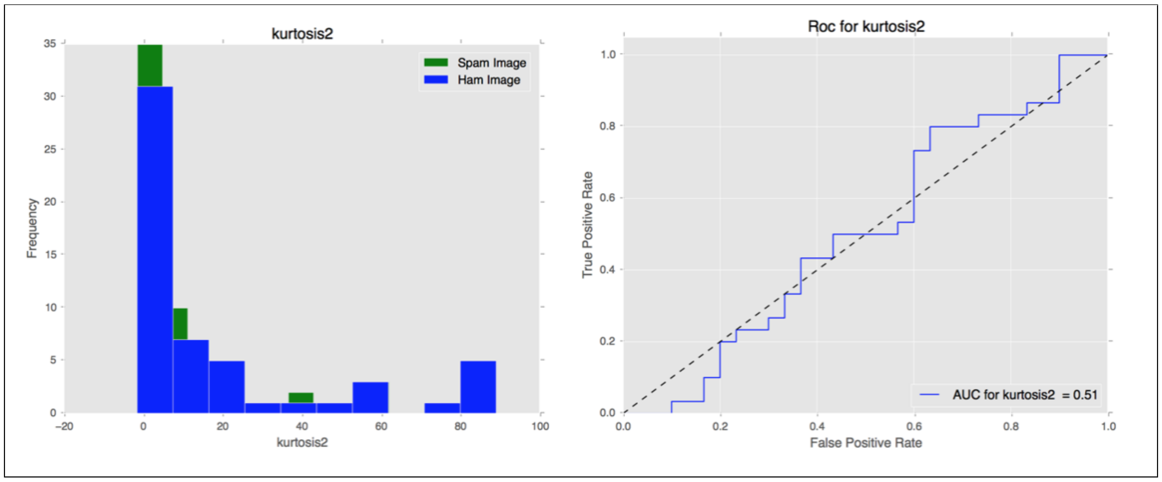


(b) Dataset 3

Figure A.59: Feature Distribution and ROC for Kurtosis of First Color Channel

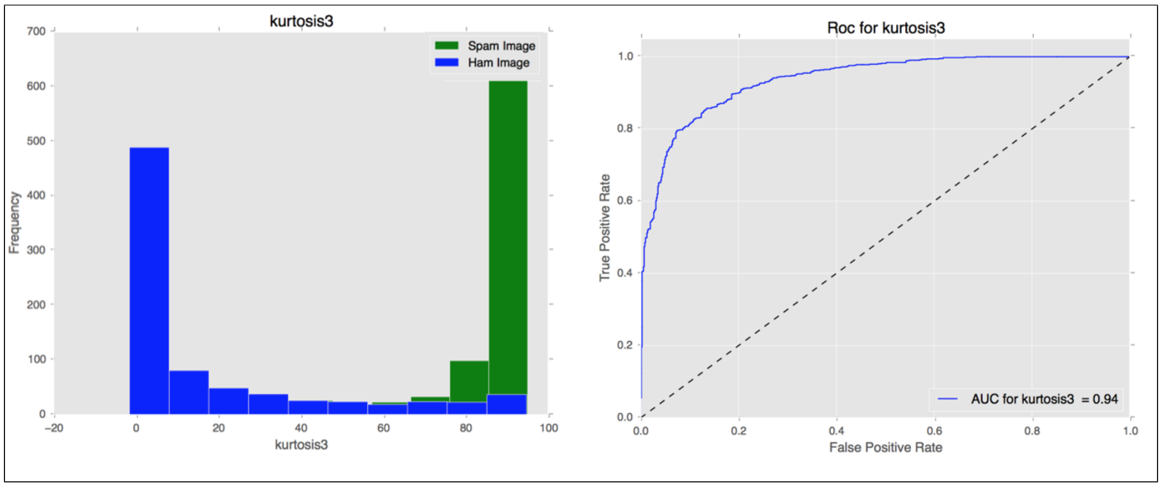


(a) Dataset 1

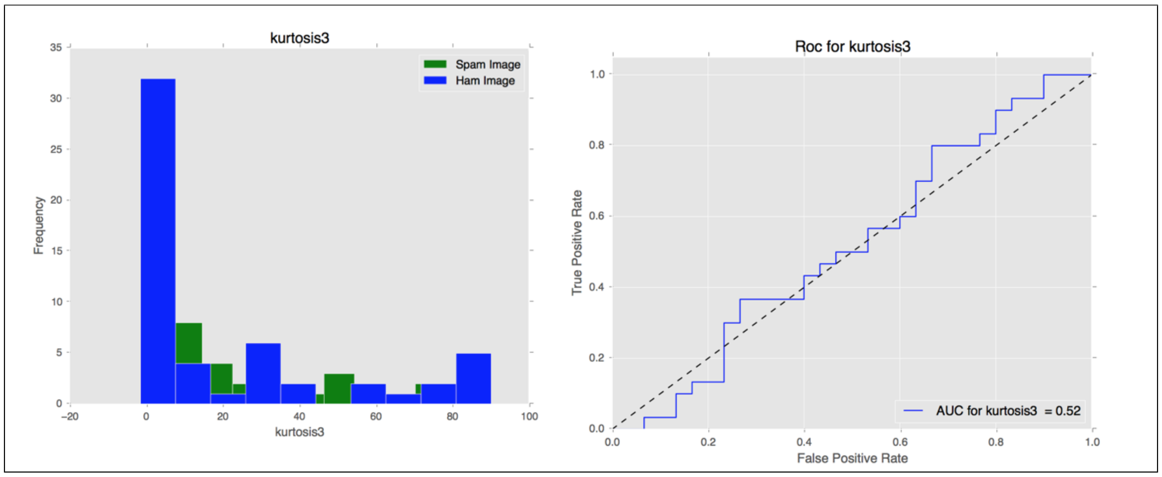


(b) Dataset 3

Figure A.60: Feature Distribution and ROC for Kurtosis of Second Color Channel

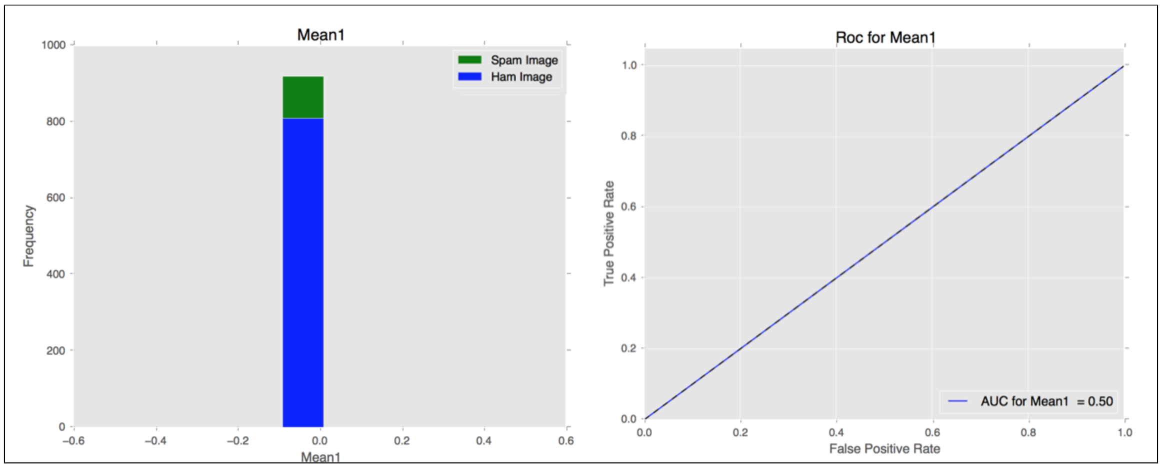


(a) Dataset 1

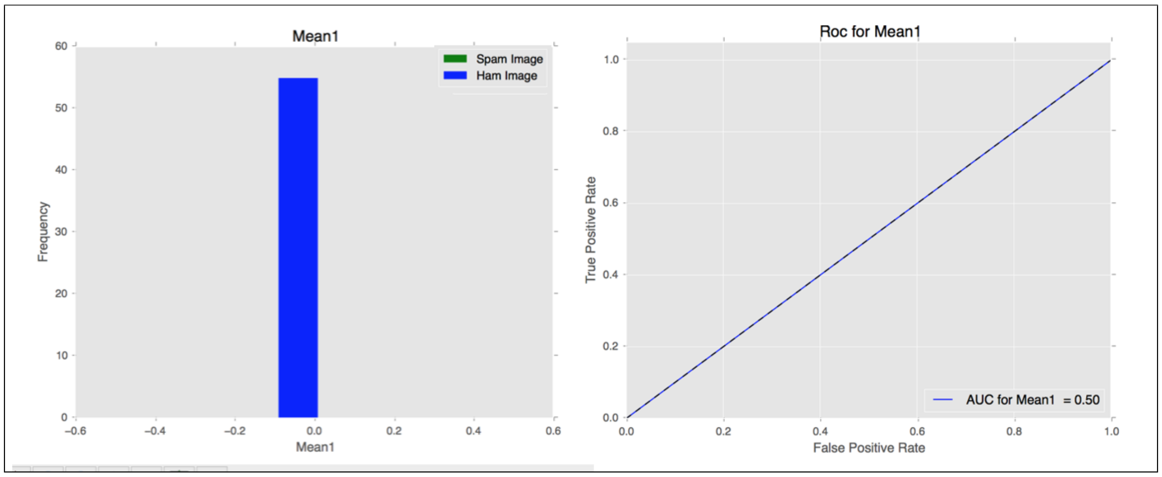


(b) Dataset 3

Figure A.61: Feature Distribution and ROC for Kurtosis of Third Color Channel

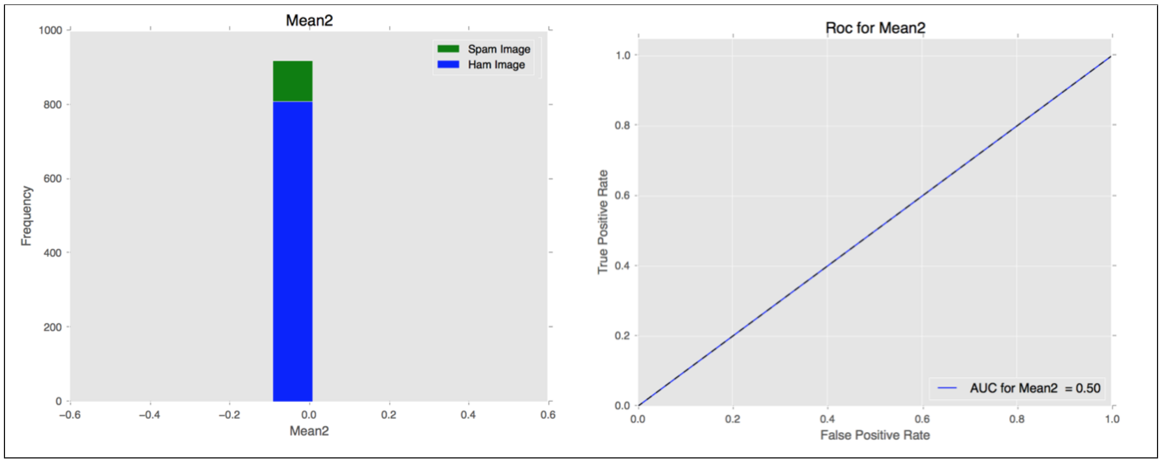


(a) Dataset 1

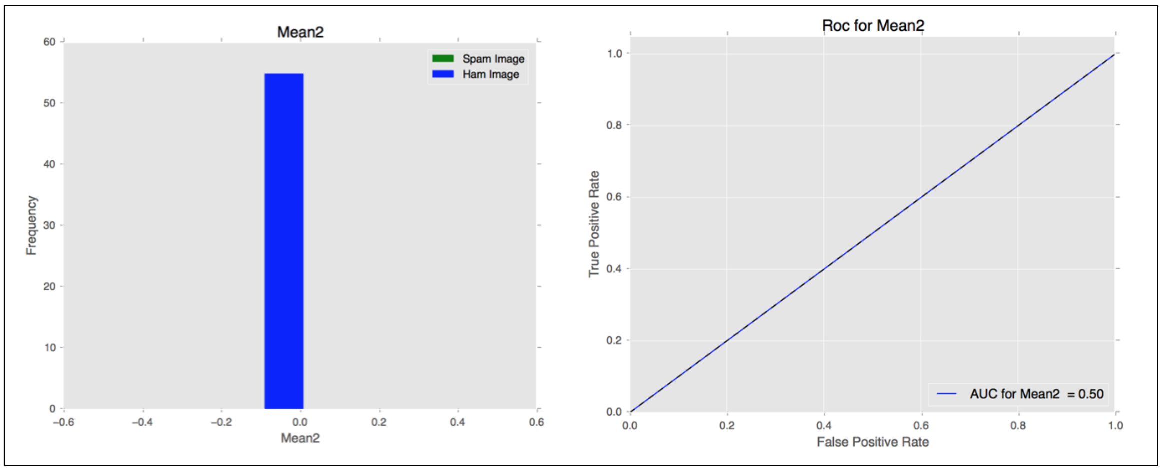


(b) Dataset 3

Figure A.62: Feature Distribution and ROC for Mean First Color Channel

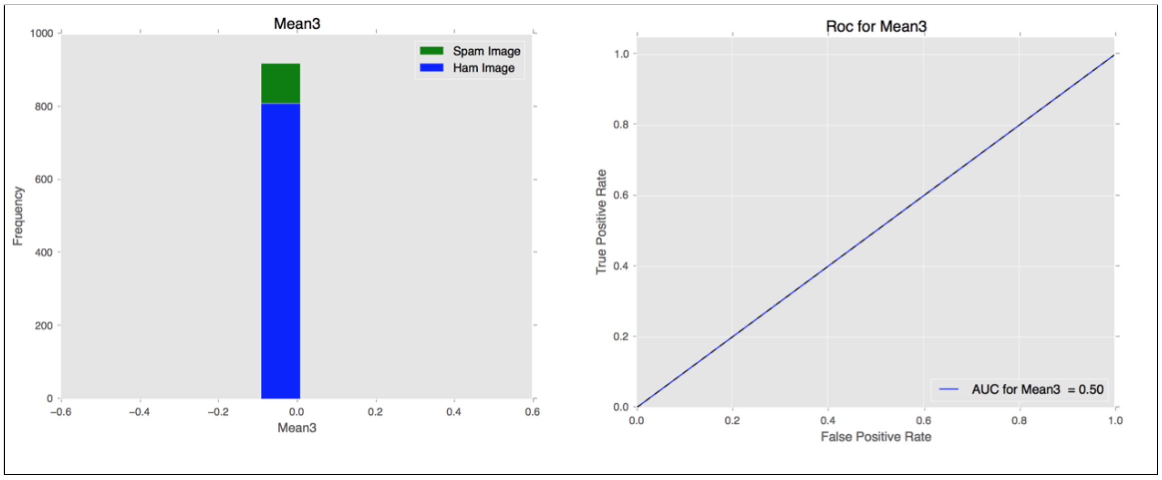


(a) Dataset 1

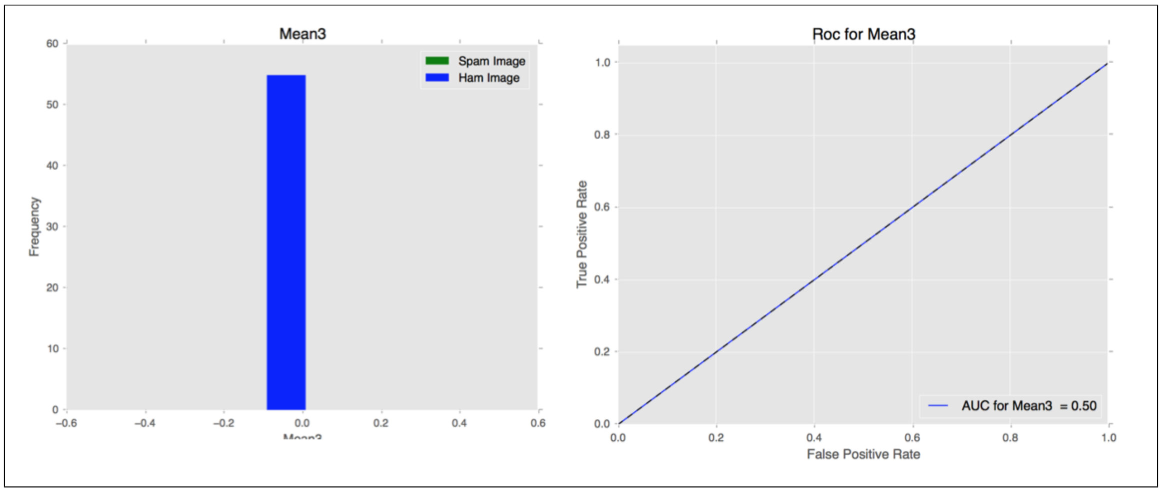


(b) Dataset 3

Figure A.63: Feature Distribution and ROC for Mean First Color Channel

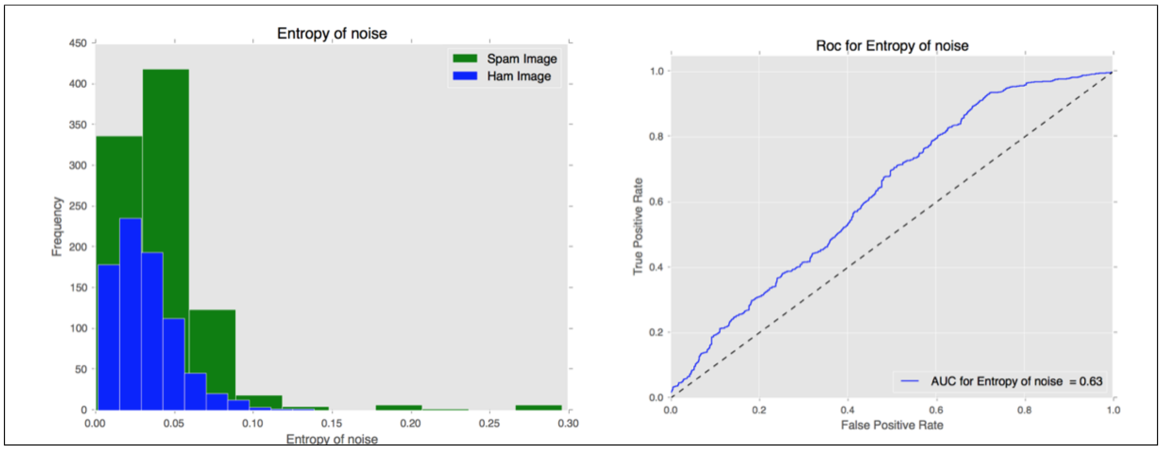


(a) Dataset 1

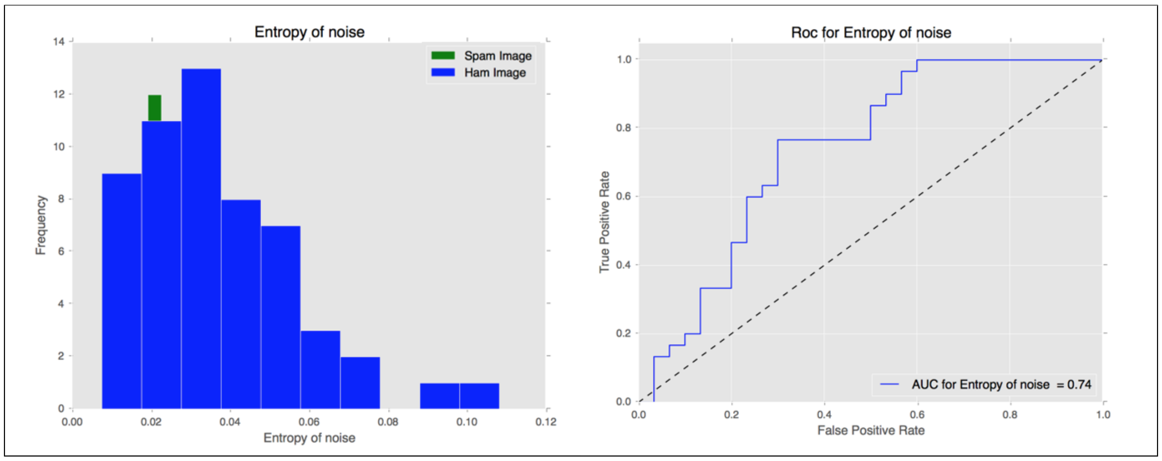


(b) Dataset 3

Figure A.64: Feature Distribution and ROC for Mean of Third Color Channel



(a) Dataset 1



(b) Dataset 3

Figure A.65: Feature Distribution and ROC for Entropy of Noise

APPENDIX B

Feature Selection Experiments

This section presents the feature selection experiment results for feature ranking method and RFE method as a part of SVM-based detection.

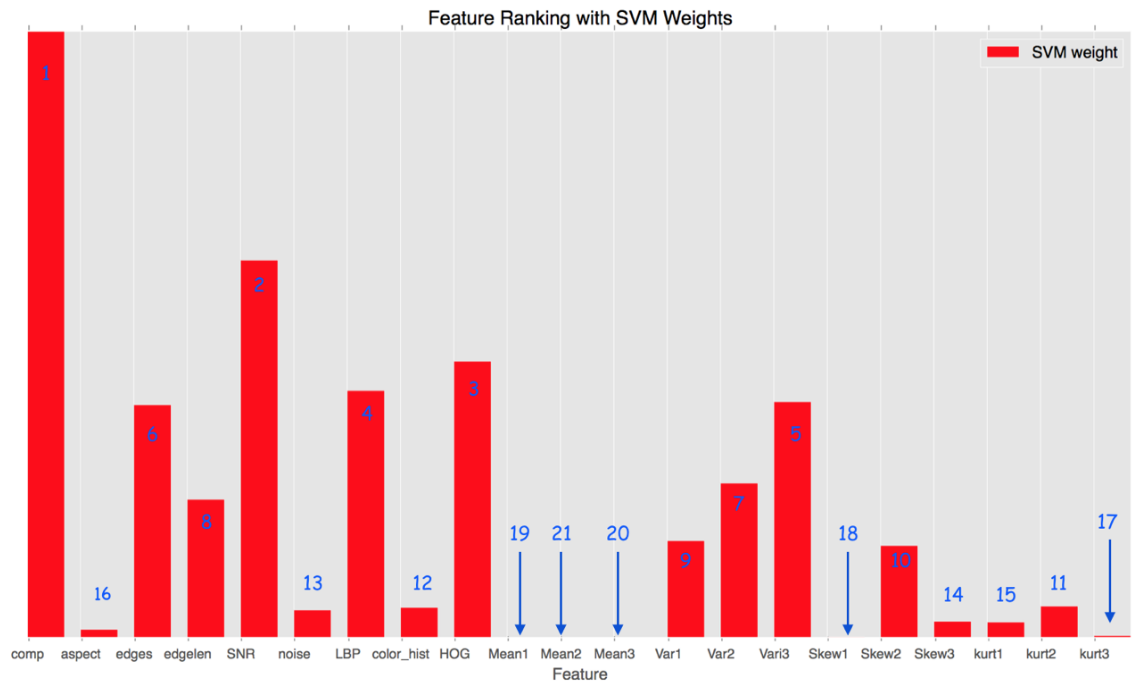


Figure B.66: SVM Weight Ranking

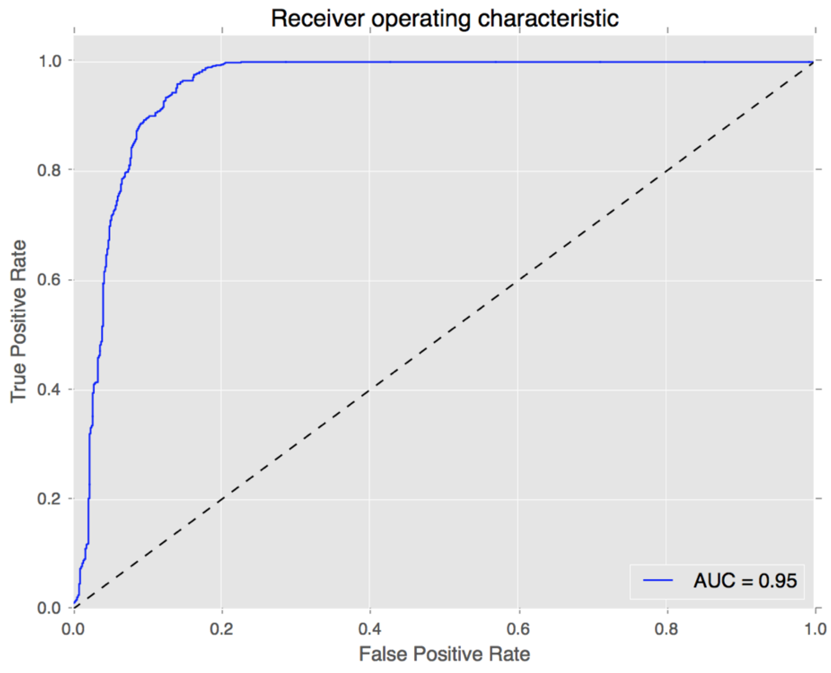


Figure B.67: ROC for SVM Ranking Feature Selection 1 Feature Subset

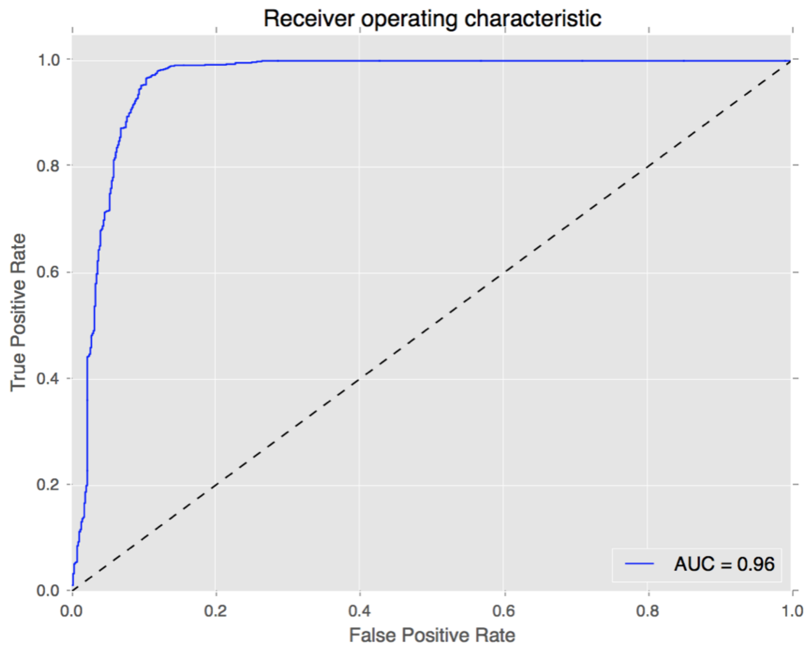


Figure B.68: ROC for SVM Ranking Feature Selection 2 Feature Subset

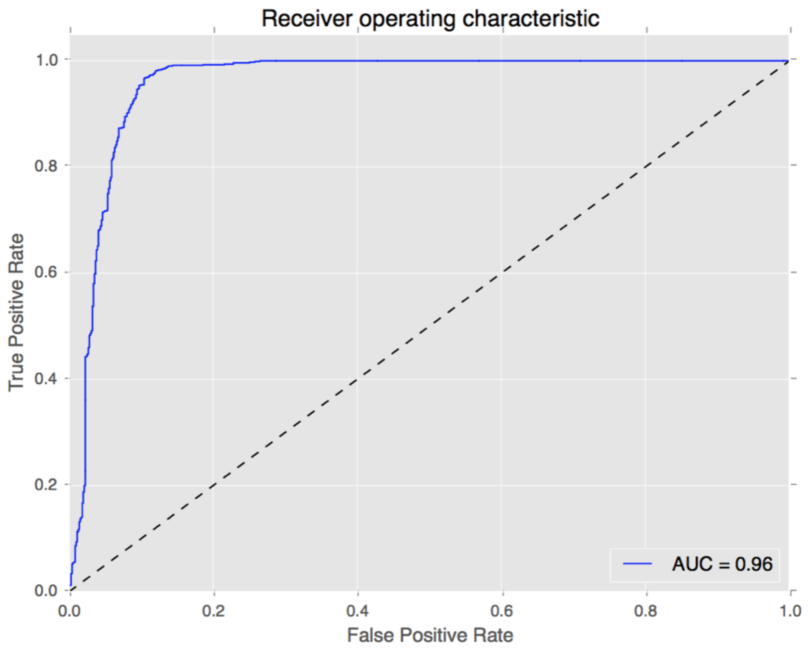


Figure B.69: ROC for SVM Ranking Feature Selection 3 Feature Subset

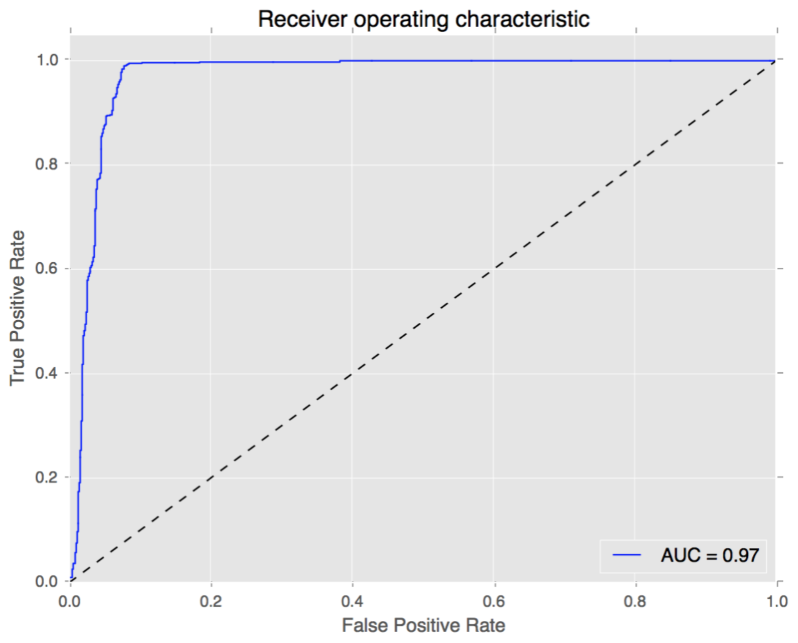


Figure B.70: ROC for SVM Ranking Feature Selection 4 Feature Subset

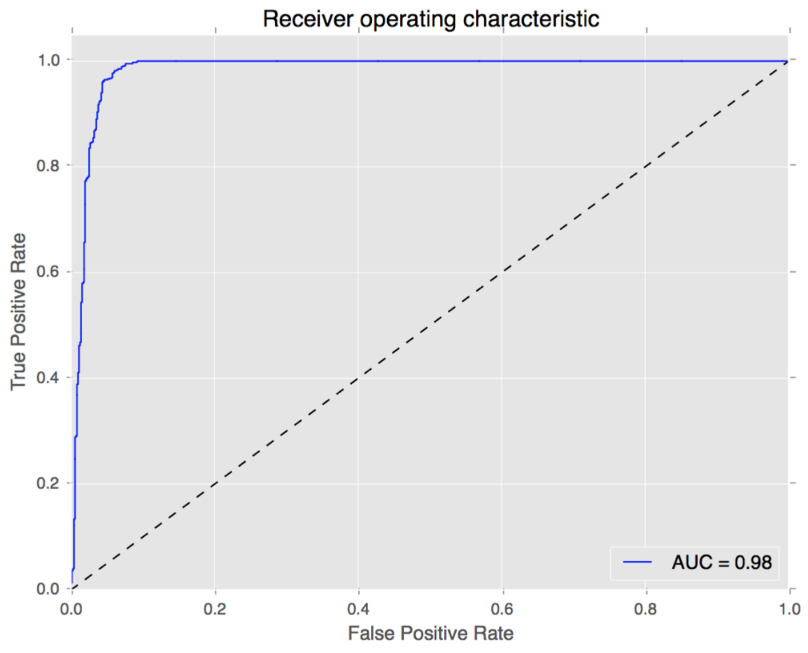


Figure B.71: ROC for SVM Ranking Feature Selection 5 Feature Subset

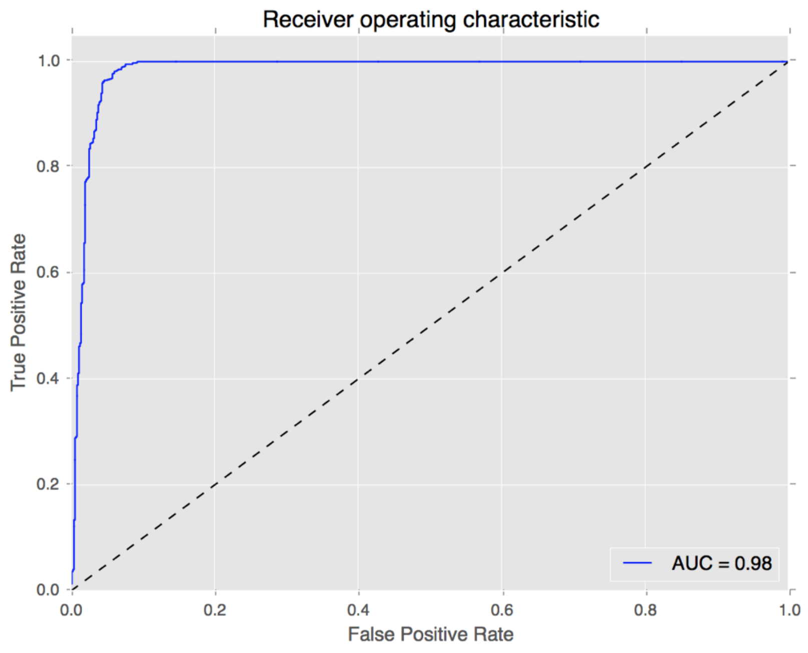


Figure B.72: ROC for SVM Ranking Feature Selection 6 Feature Subset

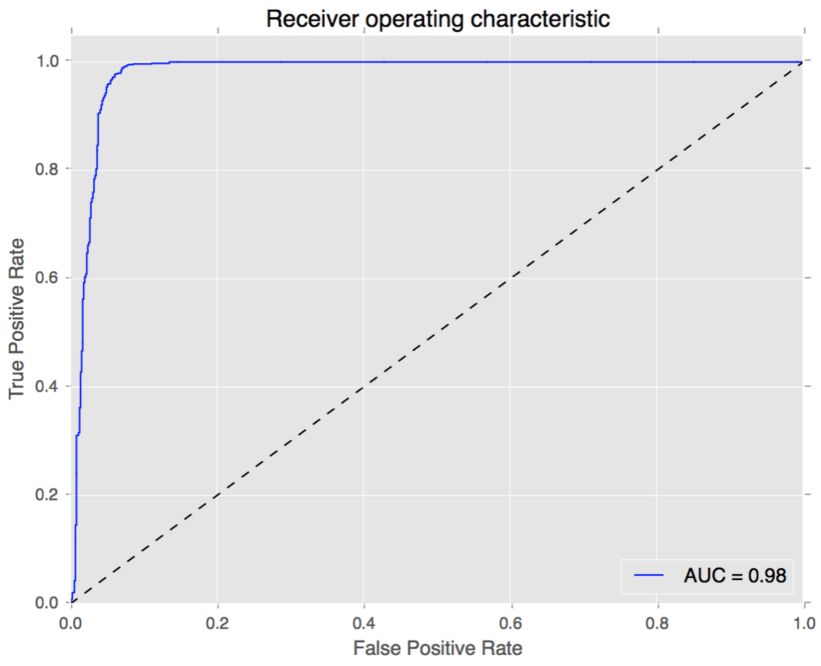


Figure B.73: ROC for SVM Ranking Feature Selection 7 Feature Subset

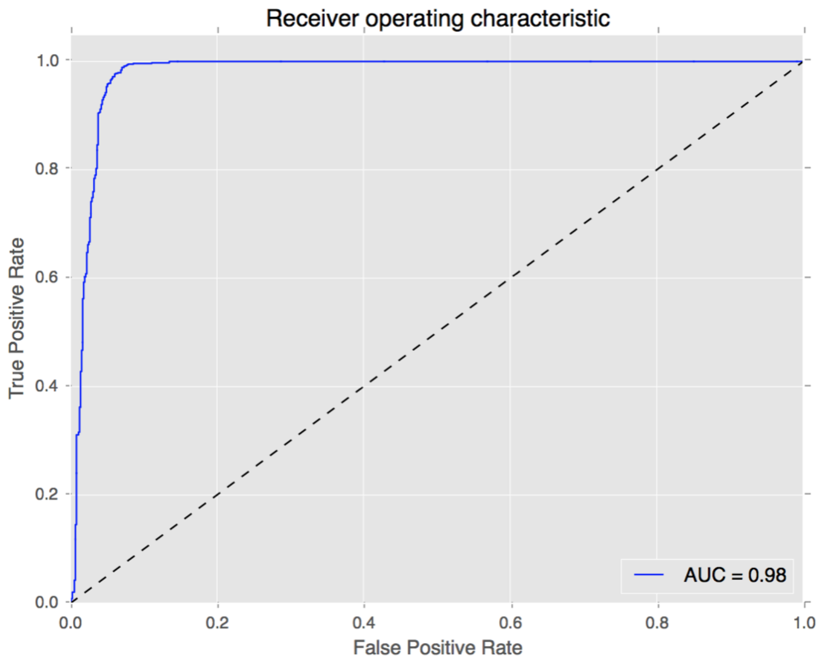


Figure B.74: ROC for SVM Ranking Feature Selection 8 Feature Subset

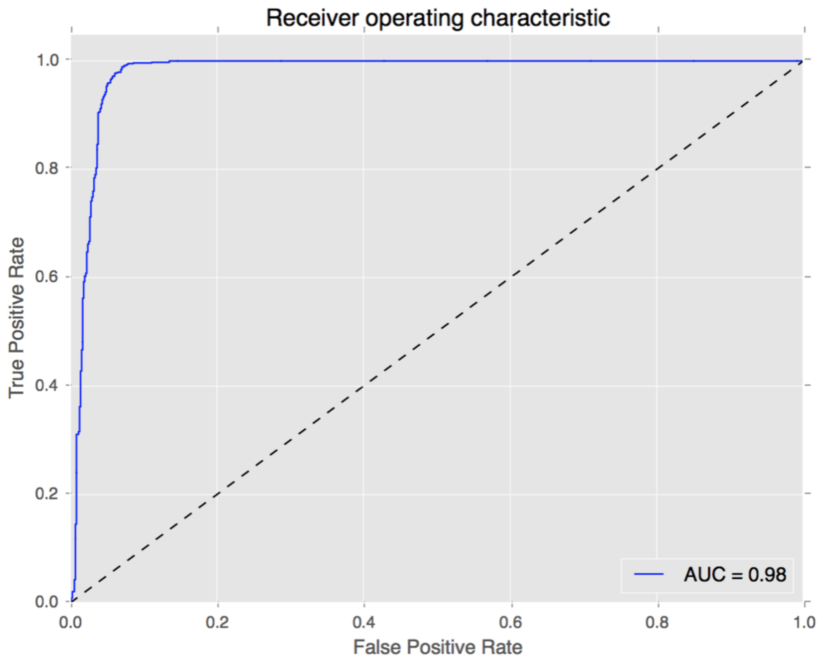


Figure B.75: ROC for SVM Ranking Feature Selection 9 Feature Subset

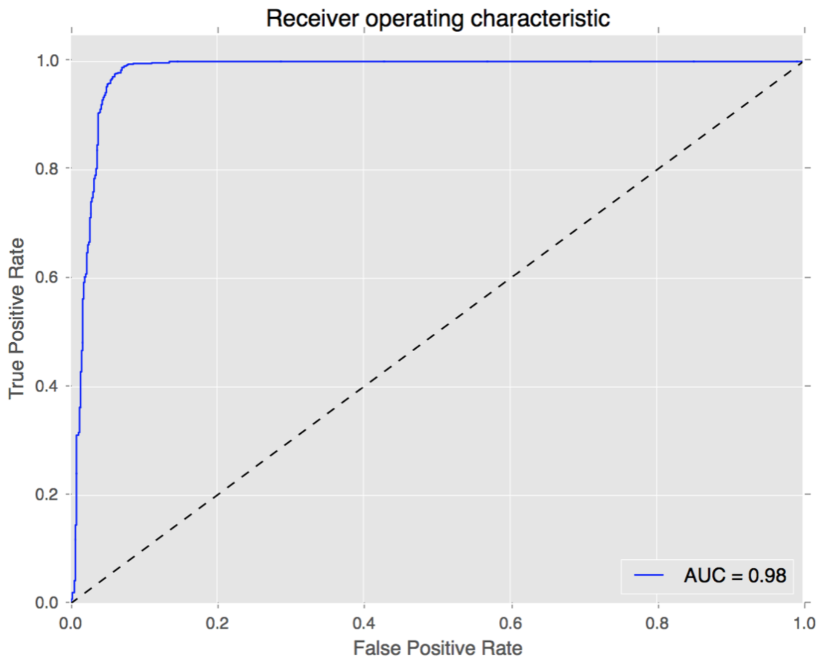


Figure B.76: ROC for SVM Ranking Feature Selection 10 Feature Subset

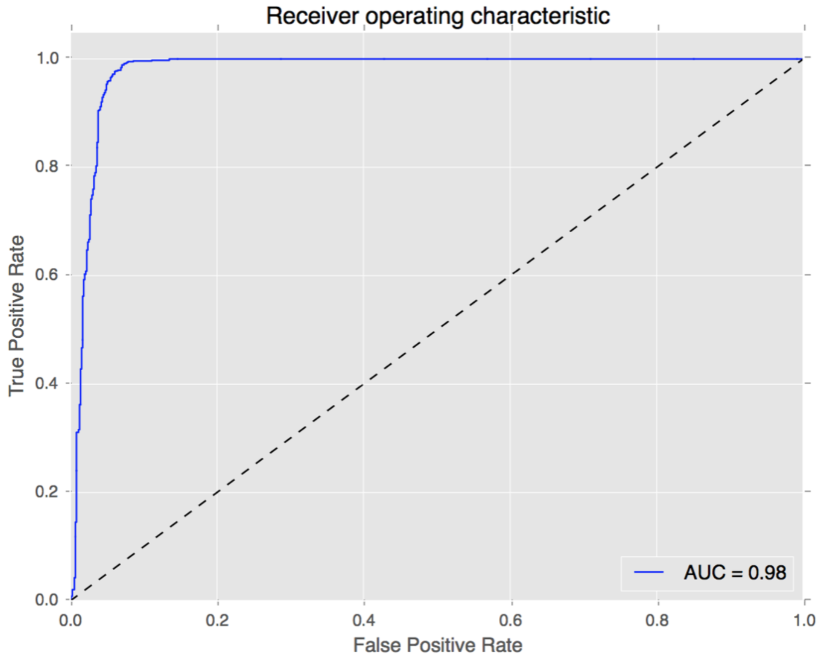


Figure B.77: ROC for SVM Ranking Feature Selection 11 Feature Subset

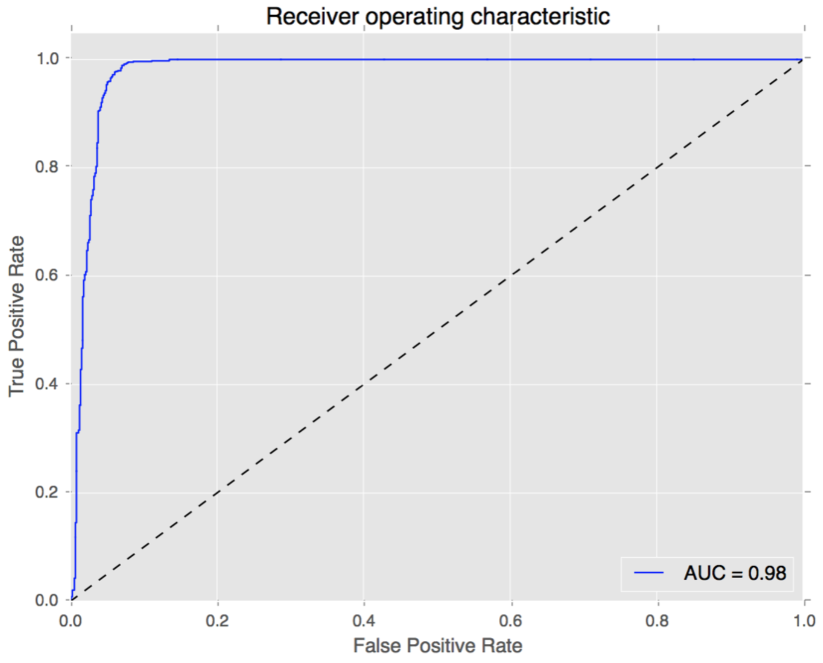


Figure B.78: ROC for SVM Ranking Feature Selection 12 Feature Subset

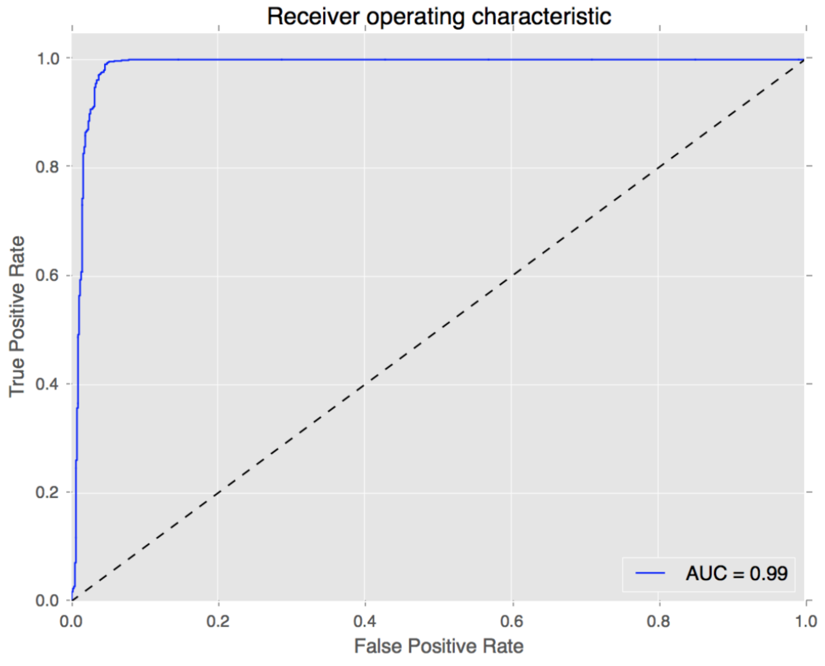


Figure B.79: ROC for SVM Ranking Feature Selection 13 Feature Subset

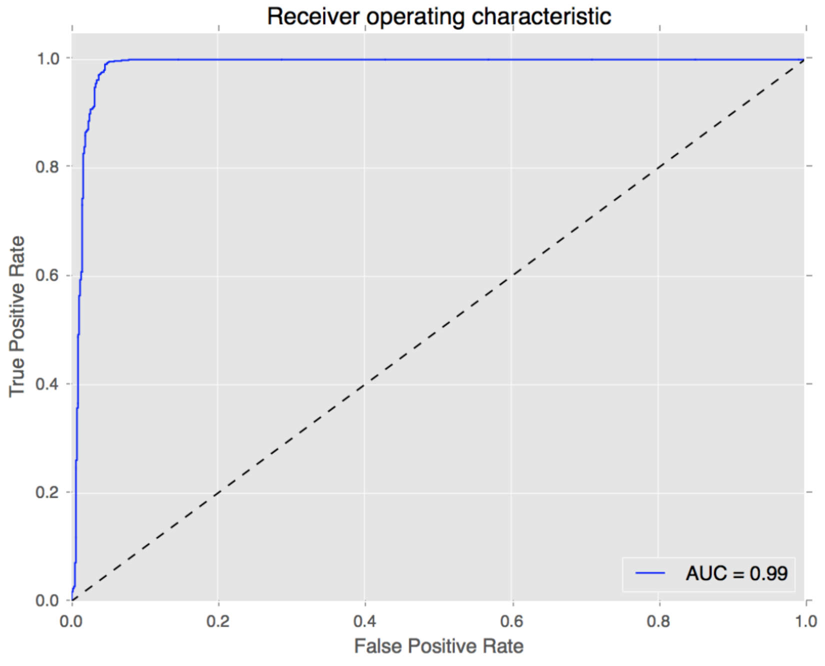
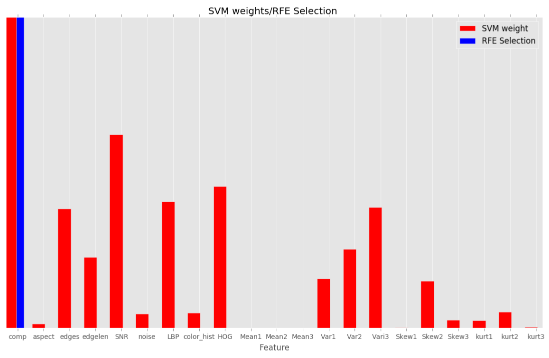
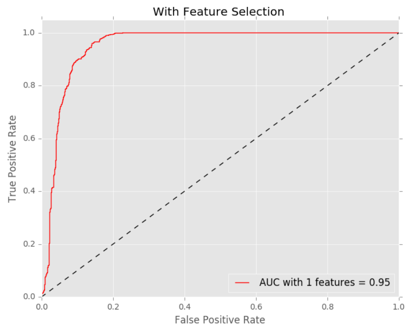


Figure B.80: ROC for SVM Ranking Feature Selection 14 Feature Subset

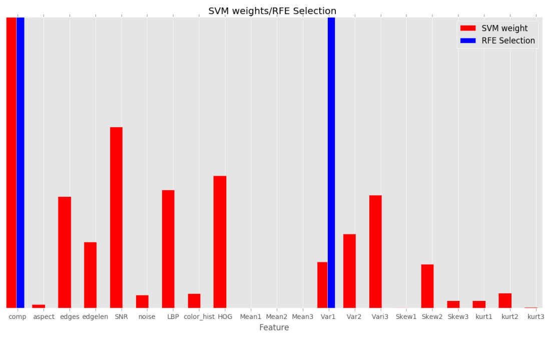


(a) RFE selection

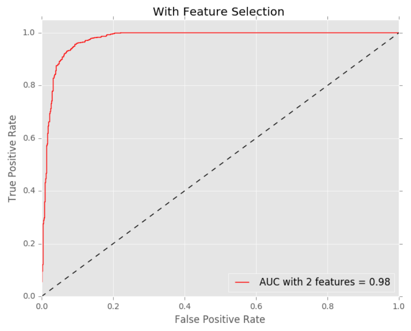


(b) ROC

Figure B.81: RFE Selection for 1 Feature Subset

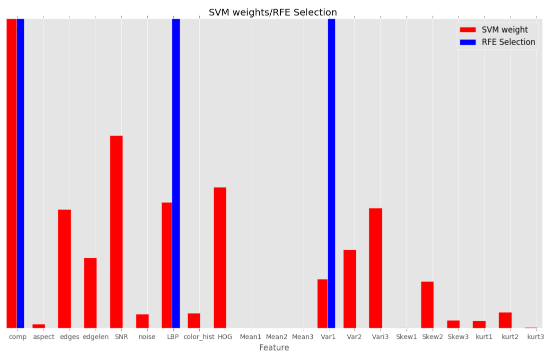


(a) RFE selection

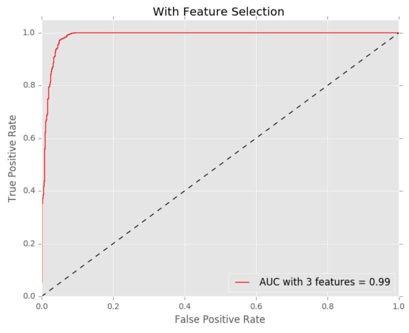


(b) ROC

Figure B.82: RFE Selection for 2 Feature Subset

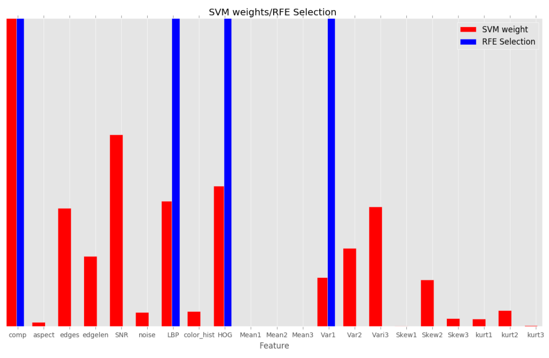


(a) RFE selection

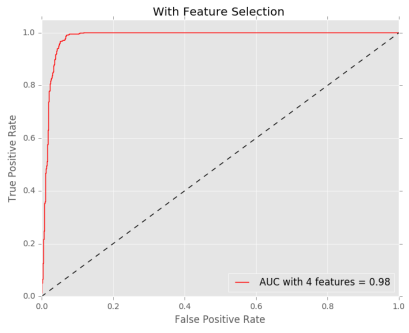


(b) ROC

Figure B.83: RFE Selection for 3 Feature Subset

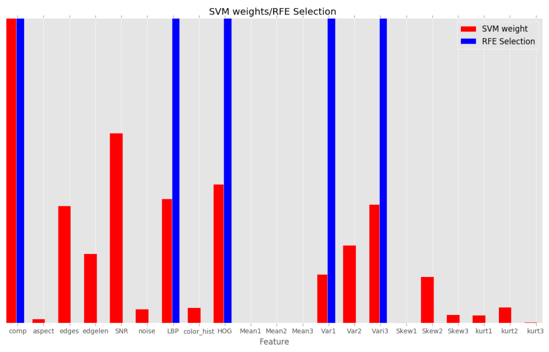


(a) RFE selection

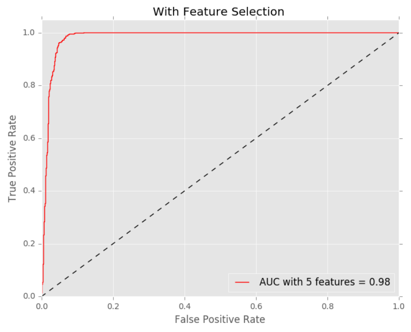


(b) ROC

Figure B.84: RFE Selection for 4 Feature Subset

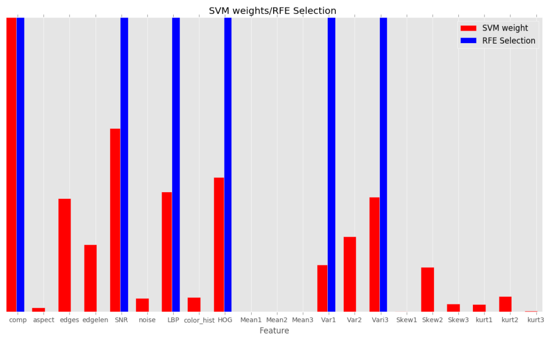


(a) RFE selection

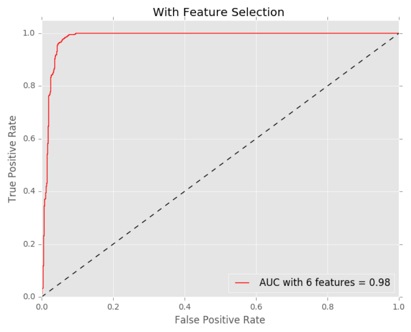


(b) ROC

Figure B.85: RFE Selection for 5 Feature Subset

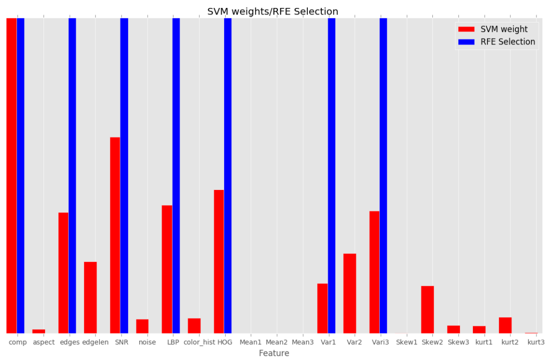


(a) RFE selection

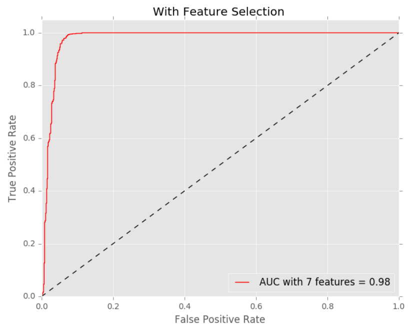


(b) ROC

Figure B.86: RFE Selection for 6 Feature Subset

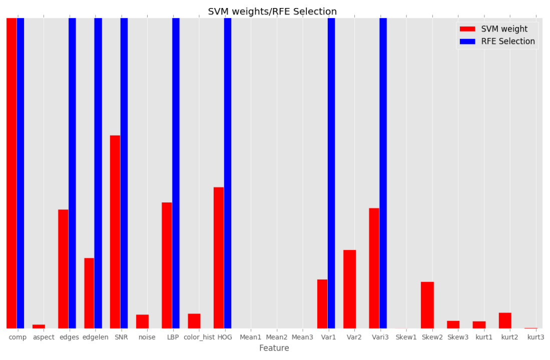


(a) RFE selection

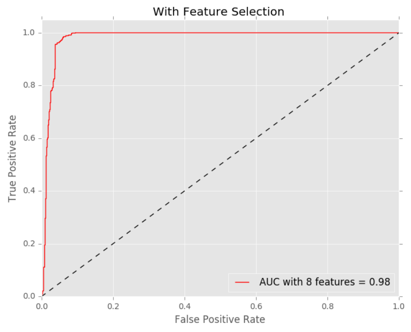


(b) ROC

Figure B.87: RFE Selection for 7 Feature Subset

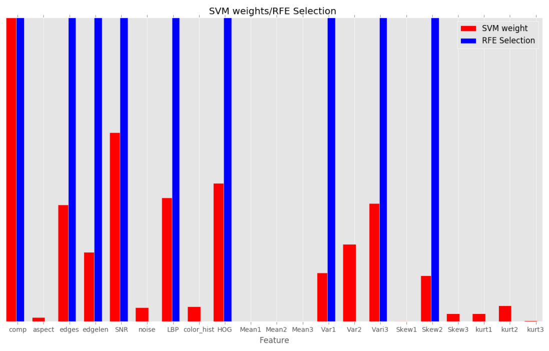


(a) RFE selection

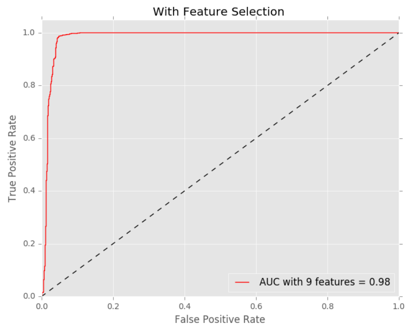


(b) ROC

Figure B.88: RFE Selection for 8 Feature Subset

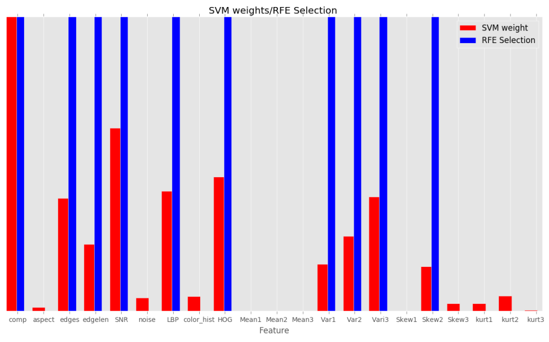


(a) RFE selection

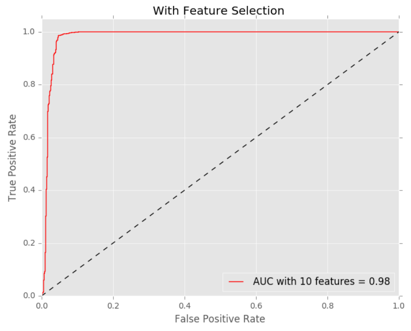


(b) ROC

Figure B.89: RFE Selection for 9 Feature Subset

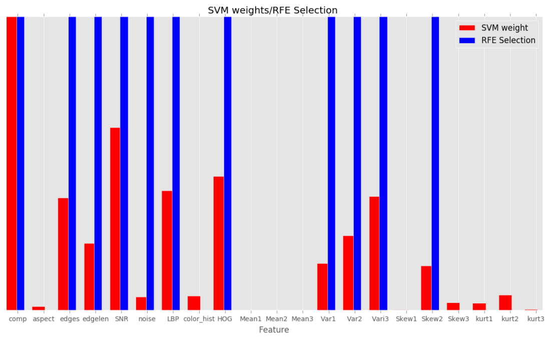


(a) RFE selection

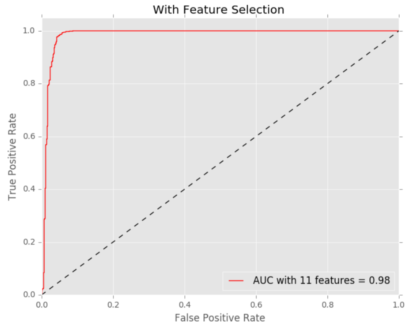


(b) ROC

Figure B.90: RFE Selection for 10 Feature Subset

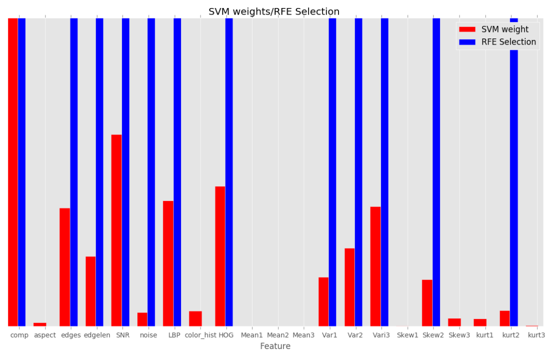


(a) RFE selection

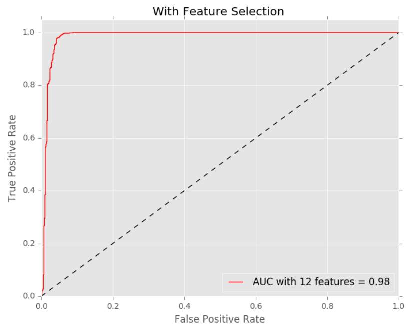


(b) ROC

Figure B.91: RFE Selection for 11 Feature Subset

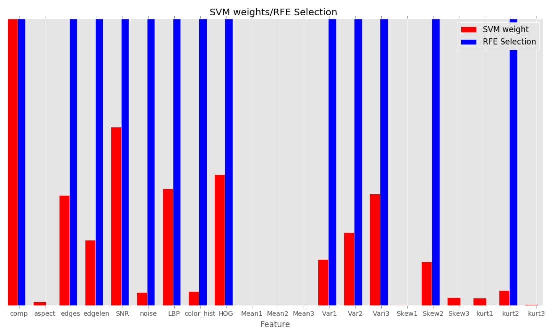


(a) RFE selection

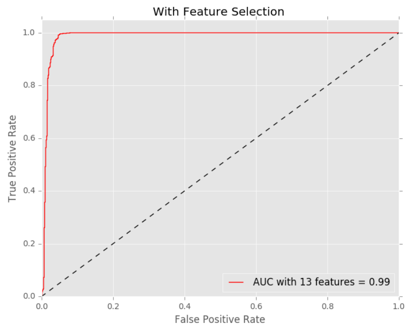


(b) ROC

Figure B.92: RFE Selection for 12 Feature Subset

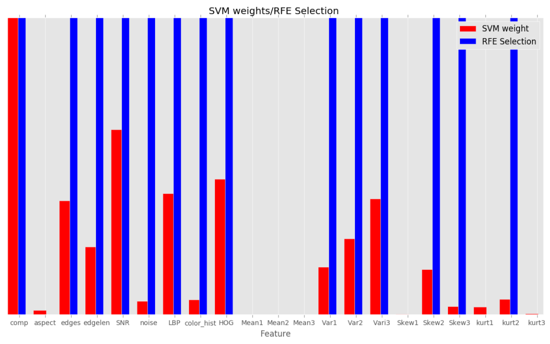


(a) RFE selection

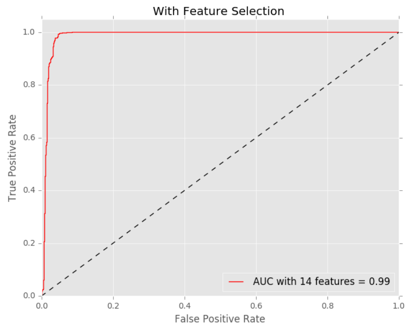


(b) ROC

Figure B.93: RFE Selection for 13 Feature Subset

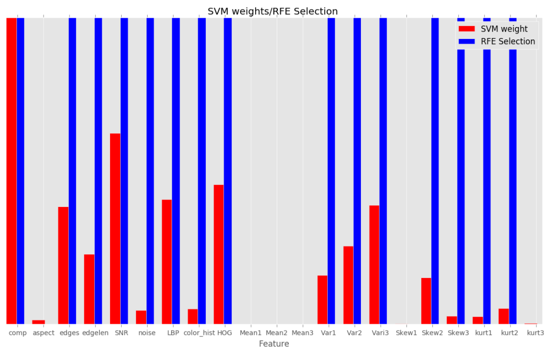


(a) RFE selection

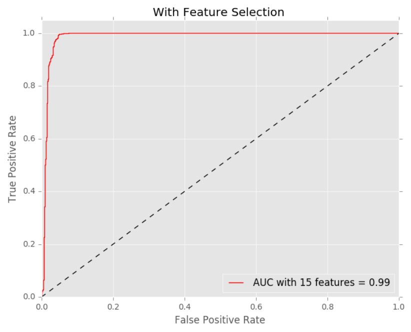


(b) ROC

Figure B.94: RFE Selection for 14 Feature Subset

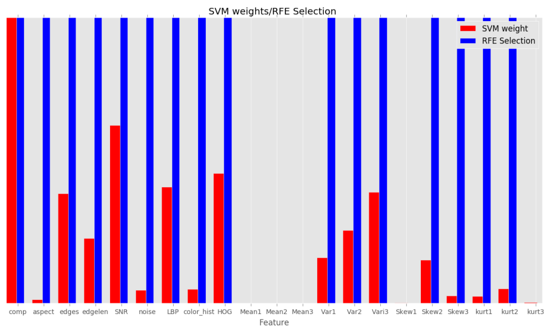


(a) RFE selection

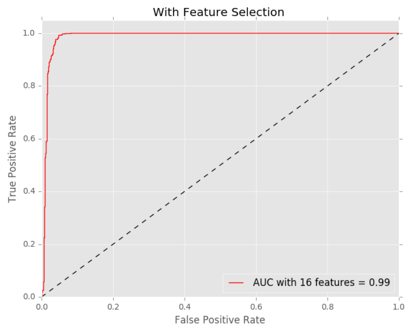


(b) ROC

Figure B.95: RFE Selection for 15 Feature Subset

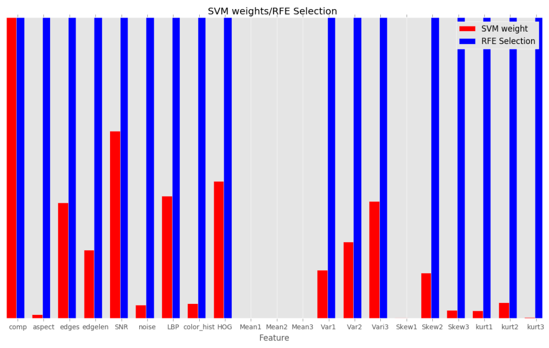


(a) RFE selection

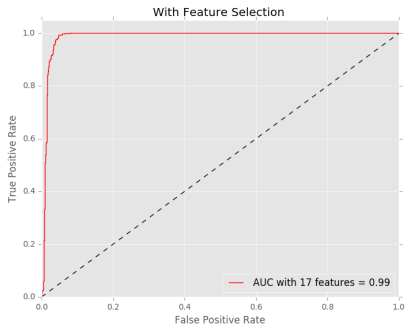


(b) ROC

Figure B.96: RFE Selection for 16 Feature Subset

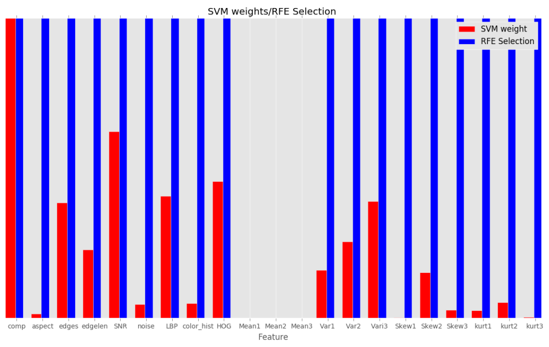


(a) RFE selection

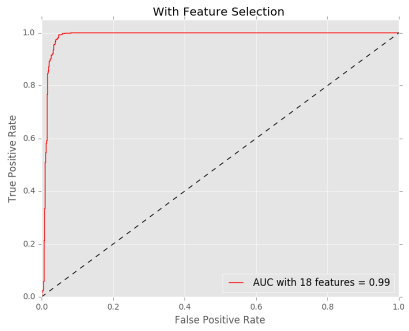


(b) ROC

Figure B.97: RFE Selection for 17 Feature Subset

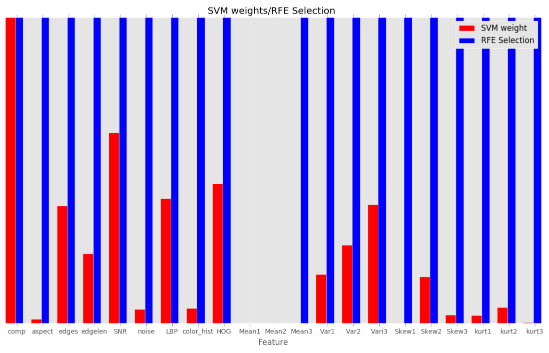


(a) RFE selection

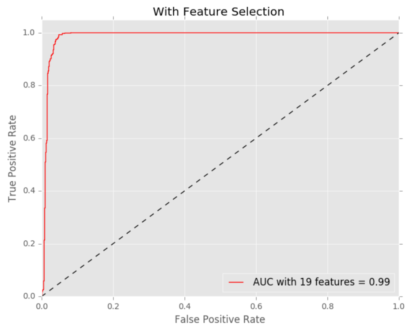


(b) ROC

Figure B.98: RFE Selection for 18 Feature Subset

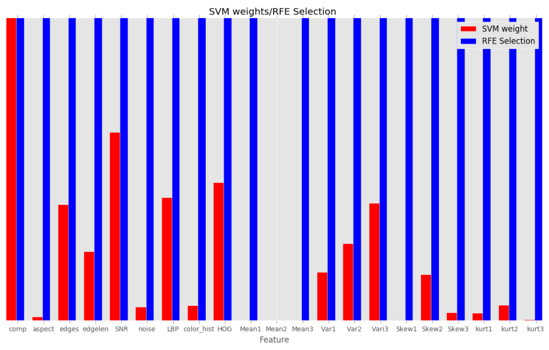


(a) RFE selection

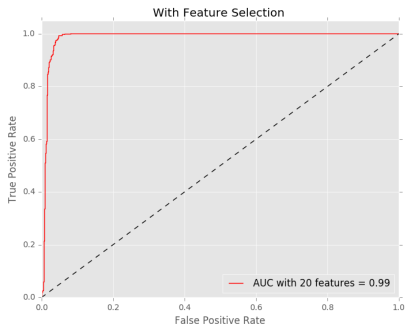


(b) ROC

Figure B.99: RFE Selection for 19 Feature Subset



(a) RFE selection



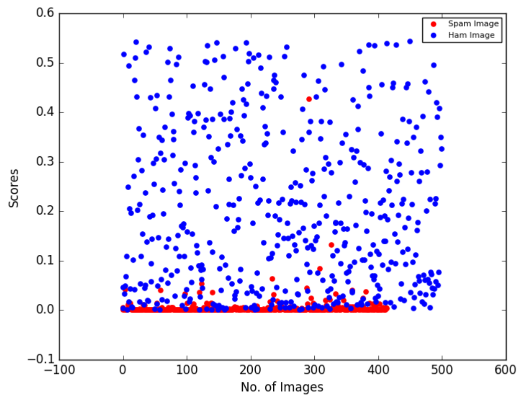
(b) ROC

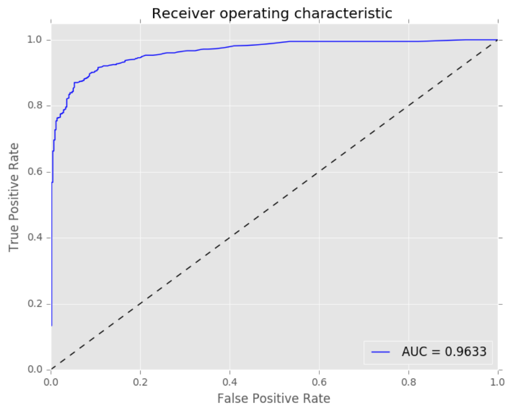
Figure B.100: RFE Selection for 20 Feature Subset

APPENDIX C

PCA-based Experiments

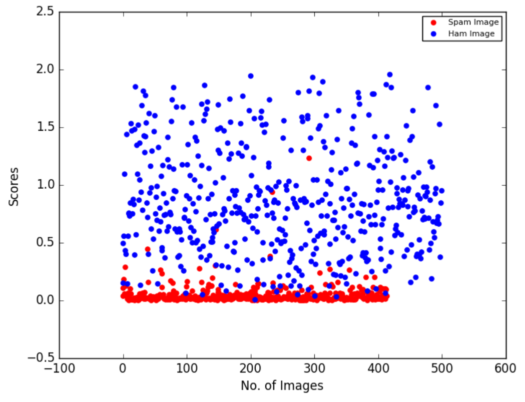
In this section, we present the scatterplots and the ROC curves of the scores obtained for the increasing value of eigen component. The AUC values from these curves have been used to plot the Figure 34.



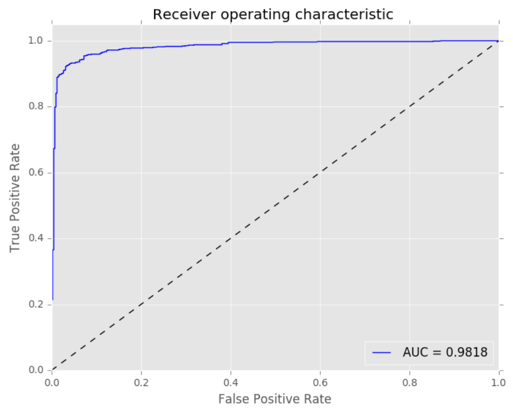


(a) (b)

Figure C.101: Scatterplot and ROC for Eigen Component = 1

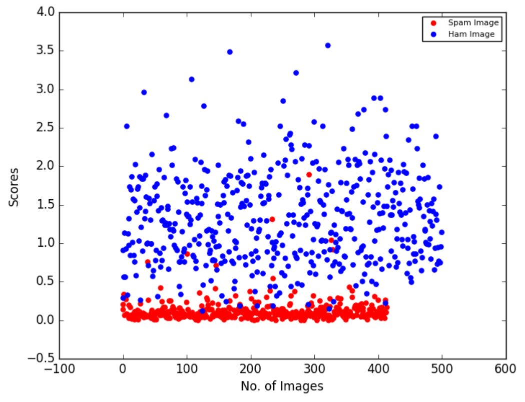


(a)

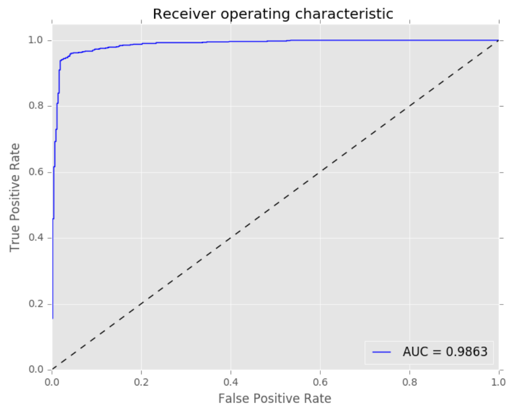


(b)

Figure C.102: Scatterplot and ROC for Eigen Component = 2

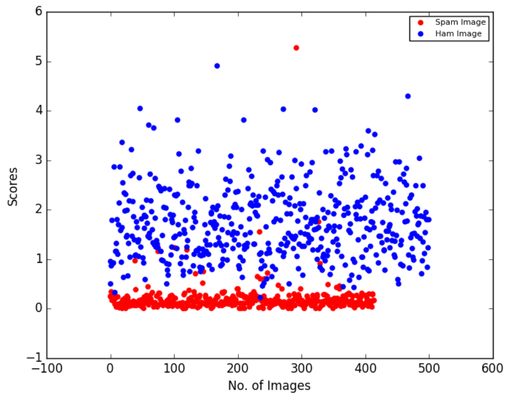


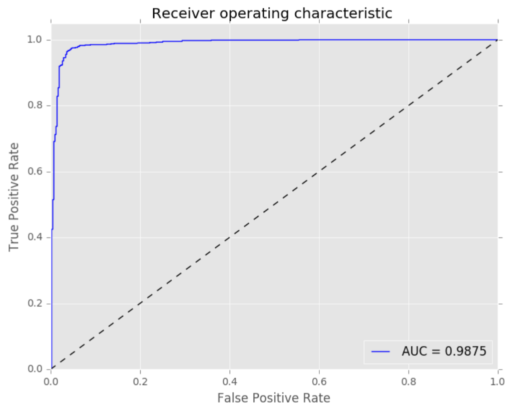
(a)



(b)

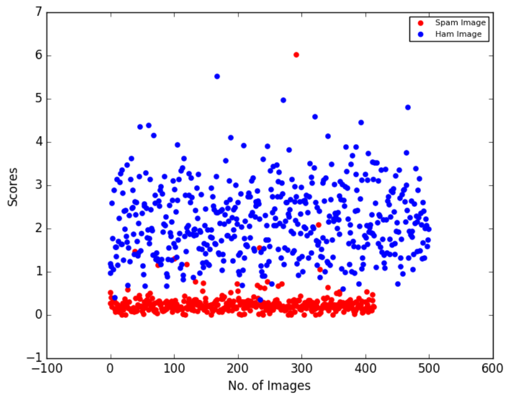
Figure C.103: Scatterplot and ROC for Eigen Component = 3

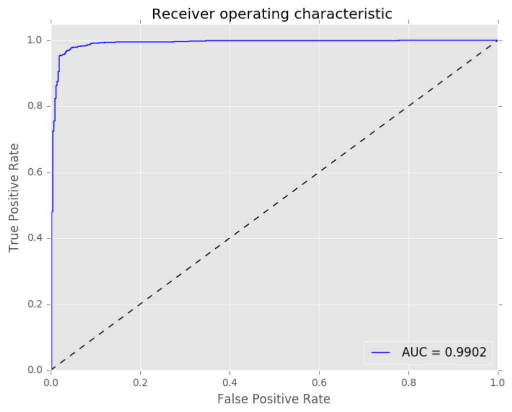




(a) (b)

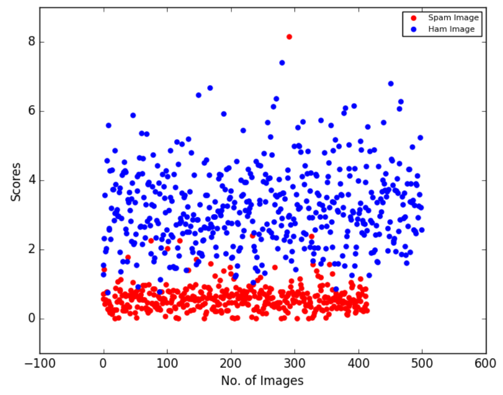
Figure C.104: Scatterplot and ROC for Eigen Component = 4



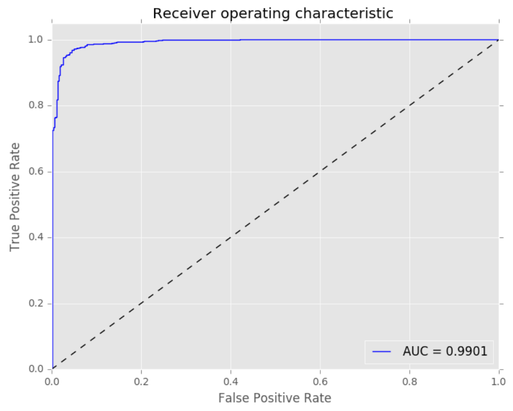


(a) (b)

Figure C.105: Scatterplot and ROC for Eigen Component = 5

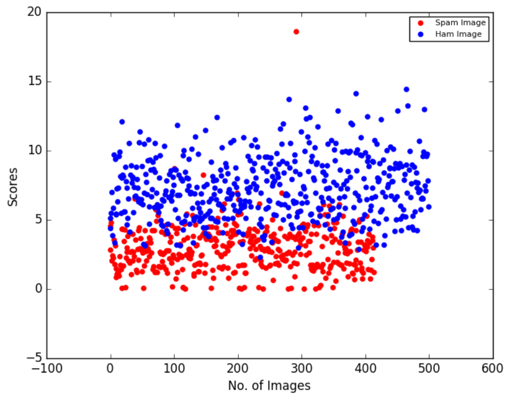


(a)

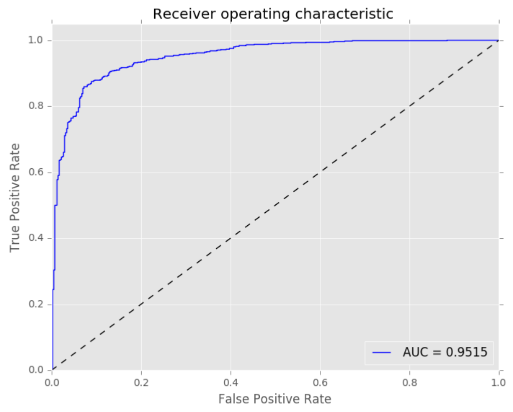


(b)

Figure C.106: Scatterplot and ROC for Eigen Component = 10

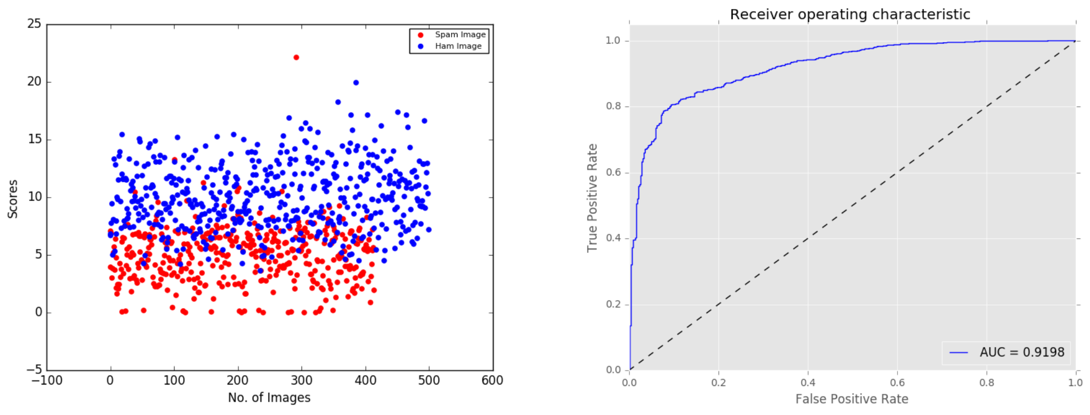


(a)



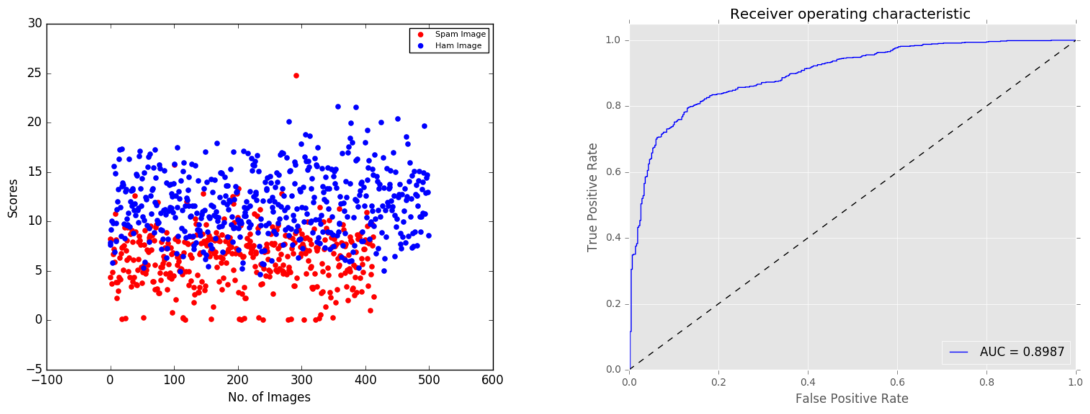
(b)

Figure C.107: Scatterplot and ROC for Eigen Component = 50



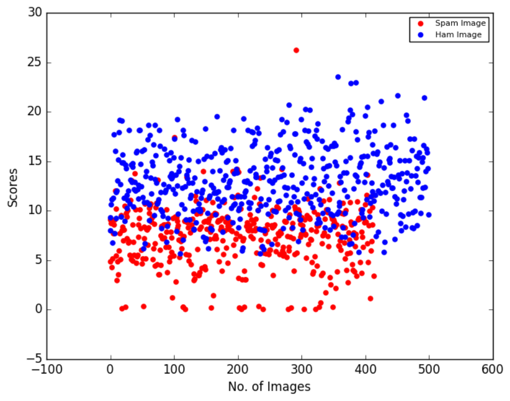
(a) (b)

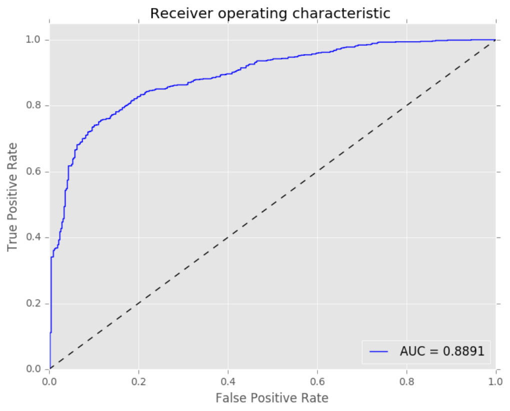
Figure C.108: Scatterplot and ROC for Eigen Component = 100



(a) (b)

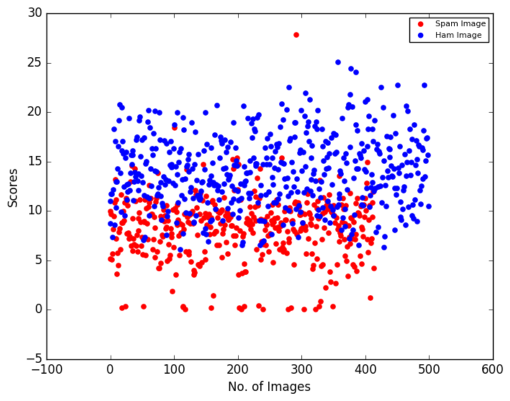
Figure C.109: Scatterplot and ROC for Eigen Component = 150

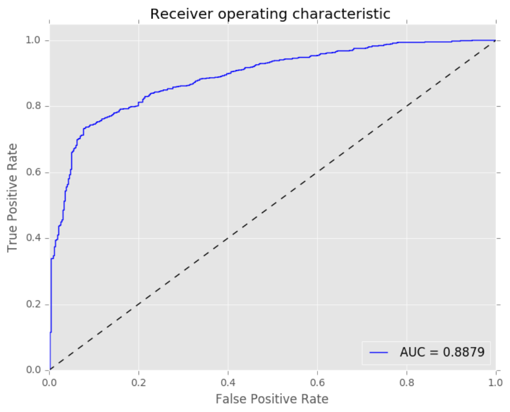




(a) (b)

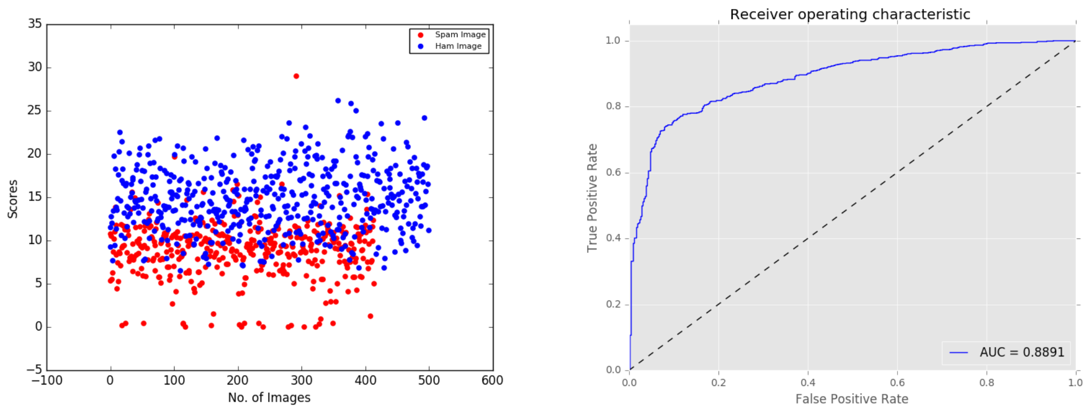
Figure C.110: Scatterplot and ROC for Eigen Component = 200





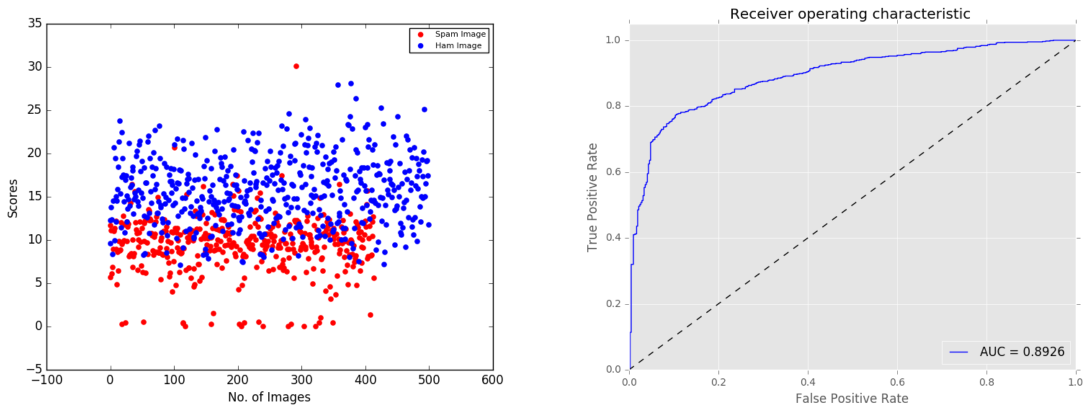
(a) (b)

Figure C.111: Scatterplot and ROC for Eigen Component = 250



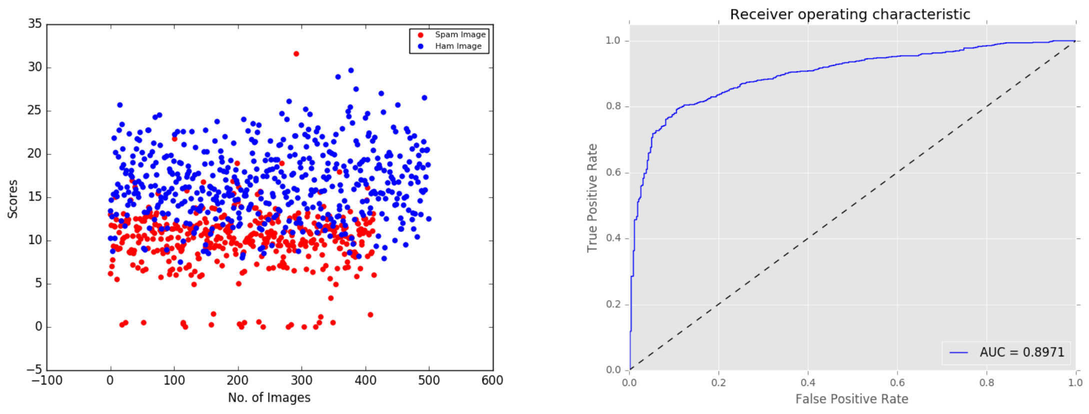
(a) (b)

Figure C.112: Scatterplot and ROC for Eigen Component = 300



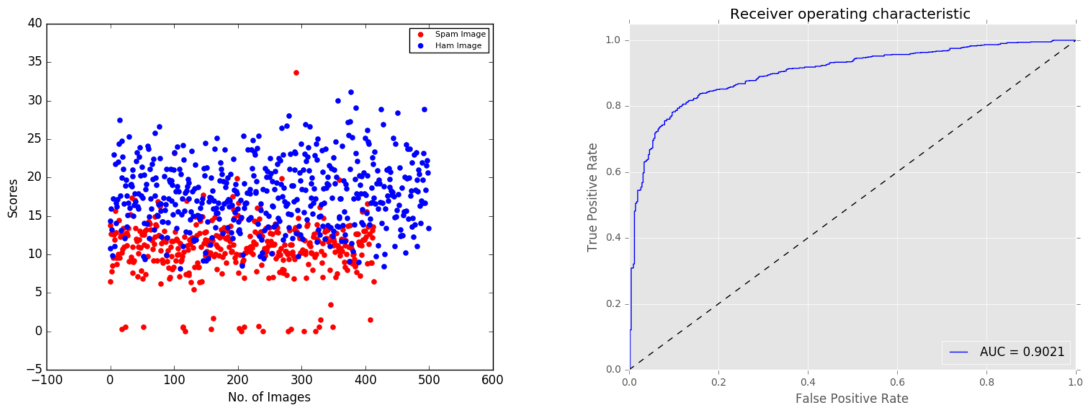
(a) (b)

Figure C.113: Scatterplot and ROC for Eigen Component = 350



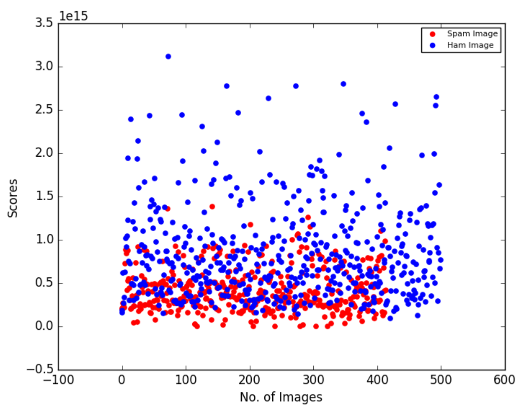
(a) (b)

Figure C.114: Scatterplot and ROC for Eigen Component = 400

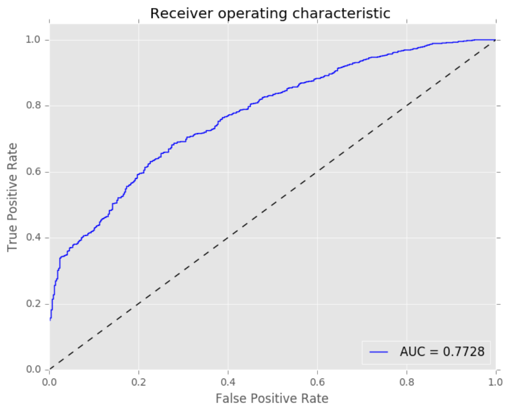


(a) (b)

Figure C.115: Scatterplot and ROC for Eigen Component = 450



(a)



(b)

Figure C.116: Scatterplot and ROC for Eigen Component = 500