Detecting Image Spam using Visual Features and Near Duplicate Detection

Bhaskar Mehta*
Google Inc.
Brandschenkestr 110
Zurich, Switzerland
bmehta@google.com

Saurabh Nangia*
IIT Guwahati
Guwahati 781039
Assam, India
s.nangia@iitg.ernet.in

Manish Gupta*
IIT Guwahati
Guwahati 781039
Assam, India
m.gupta@iitg.ernet.in

Wolfgang Nejdl
L3S Forschungszentrum
Appelstrasse 4
Hannover, Germany
nejdl@L3S.de

ABSTRACT

Email spam is a much studied topic, but even though current email spam detecting software has been gaining a competitive edge against text based email spam, new advances in spam generation have posed a new challenge: *image-based spam*. Image based spam is email which includes embedded images containing the spam messages, but in binary format. In this paper, we study the characteristics of image spam to propose two solutions for detecting image-based spam, while drawing a comparison with the existing techniques. The first solution, which uses the visual features for classification, offers an accuracy of about 98%, i.e. an improvement of at least 6% compared to existing solutions. SVMs (*Support Vector Machines*) are used to train classifiers using judiciously decided color, texture and shape features. The second solution offers a novel approach for near duplication detection in images. It involves clustering of image GMMs (*Gaussian Mixture Models*) based on the *Agglomerative Information Bottleneck* (AIB) principle, using *Jensen-Shannon divergence* (JS) as the distance measure.

Categories and Subject Descriptors

H.3 [**Information Storage And Retrieval**]: Information Search and Retrieval.; G.3 [**Mathematics of Computing**]: Probability And Statistics.

General Terms

Security

Keywords

Email spam, Image analysis, Machine learning.

*This work was done when the author was at L3S Forschungszentrum

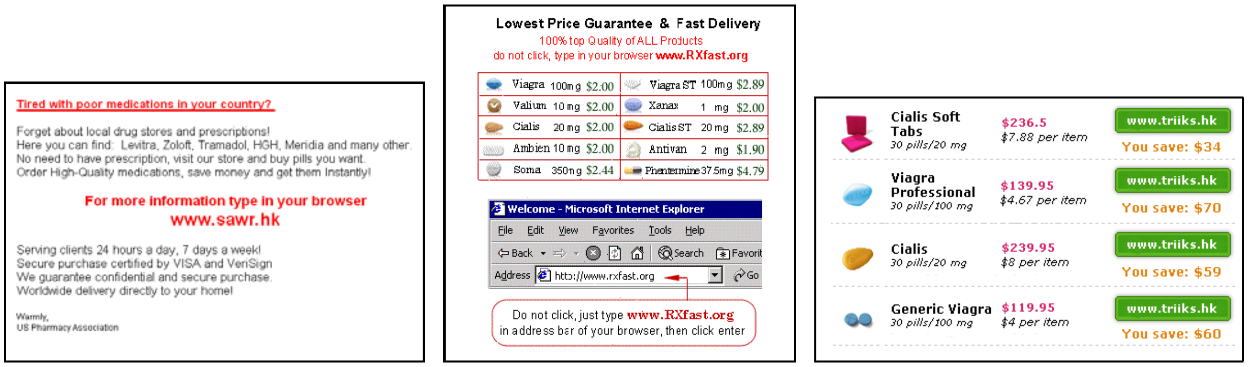
1. INTRODUCTION

Email spam has been around for more than a decade, with many covert companies offering mass emailing services for product advertisements. It has become usual for people to receive more spam email than relevant email. The design of the SMTP protocol is one main reason for email spam: it is possible to send millions of fake email messages via computer software without any authentication. Lately, authentication policies have been adopted by various web servers, where webmasters can maintain a list of blocked web domains. However, spammers keep registering millions of domain names at a pace far greater than what system administrators can cope up with.

The biggest step in the fight against spam has been the successful design of spam filtering software. Such software typically trains classifiers to identify *spam* and *ham*. For the past decade, much progress has been made in designing accurate classifiers for text emails. These classifiers are trained to use two kinds of rules: (a), rules based on connection and relay properties of the email, and (b), rules exploiting the features extracted from the content of emails. The former category of rules utilize publicly shared blacklists of servers which have been known to transmit spam emails. This information is frequently updated, requiring spammers to change domain names and servers frequently. The second type of rules use features extracted from spam text (e.g. keywords, frequent words etc) which are compared with corresponding features from ham text (normal emails) and classifiers are learnt [6]. Such classifiers have proven to be very useful and highly accurate, even though spammers continue to devise novel ways of fooling them.

2. WHAT IS IMAGE SPAM?

Image based spam is a breakthrough from the spammers' view point; it is a simple and effective way of deceiving spam filters since they can process only text. An image spam email is formatted in HTML, which usually has only non-suspicious text with an embedded image (sent either as an attachment or via the use of self referencing HTML with image data in its payload). The embedded image carries the target message and most email clients display the message



Tired with poor medications in your country?

Forget about local drug stores and prescriptions!
Here you can find: Levitra, Zoloft, Tramadol, HGH, Meridia and many other.
No need to have prescription, visit our store and buy pills you want.
Order High-Quality medications, save money and get them Instantly!

For more information type in your browser
www.sawr.hk

Serving clients 24 hours a day, 7 days a week!
Secure purchase certified by VISA and VeriSign
We guarantee confidential and secure purchase.
Worldwide delivery directly to your home!

Warmly,
US Pharmacy Association

Lowest Price Guarantee & Fast Delivery
100% top Quality of ALL Products
do not click, type in your browser www.RXfast.org

Viagra	100mg	$2.00	Viagra ST	100mg	$2.89
Valium	10 mg	$2.00	Xanax	1 mg	$2.00
Cialis	20 mg	$2.00	Cialis ST	20 mg	$2.89
Ambien	10 mg	$2.00	Antivan	2 mg	$1.90
Soma	350mg	$2.44	Phentermine	37.5mg	$4.79

Welcome - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorit
Address http://www.rxfast.org
Go

Do not click, just type www.RXfast.org
in address bar of your browser, then click enter

Cialis Soft Tabs 30 pills/20 mg	$236.5 $7.88 per item	www.triiks.hk	You save: $34
Viagra Professional 30 pills/100 mg	$139.95 $4.67 per item	www.triiks.hk	You save: $70
Cialis 30 pills/20 mg	$239.95 $8 per item	www.triiks.hk	You save: $59
Generic Viagra 30 pills/100 mg	$119.95 $4 per item	www.triiks.hk	You save: $60

Figure 1: Examples of Image Spam. Notice the high amount of text used in combination with pictures used to make images resemble web pages.

in their entirety. Since many ham emails also have similar properties (using HTML, carrying embedded images, with normal text) as image-based emails, existing spam filters can no longer distinguish between image-based spam and image ham. A whitepaper released in November 2006 [12] shows the rise of image spam from 10% in April to 27% of all email spam in October 2006 totaling *48 billion emails everyday.* It is thus imperative to devise new approaches for identifying image-based spam in email traffic which are both fast and accurate. Clearly, identification of image-based email spam (referred as *I-spam* here onwards) requires content analysis of the attached image. Since images are a large and heterogeneous data format, it is essential to understand characteristic properties of I-spam in order to identify them successfully. I-spam has been studied very recently by some researchers and it is possible to identify some common traits of currently visible spam images; exploiting these properties forms the basis of our solution to detection of I-spam.

- *I-spam contains text messages*: Almost all images in spam emails contain text messages conveying the intent of the spammer. This text is usually an advertisement and often contains text which has been blacklisted by spam filters (e.g. Cialis pills, drug store, stock tip, etc).

- *I-spam is usually noisy and different from one another*: It takes significant effort to design a spam image; however spammers reduce the involved effort by generating spam images using algorithms which ensure each *generating* image is distinct. To achieve this, techniques used include rearranging the items in spam emails, adding random noise, changing background or font colors, using different fonts and sizes, using random patterns like lines or circles, adding borders and so on. While such changes might go unnoticed by a human at the first glance, they cause significant changes in the properties of images, resulting in statistically different data. This *obfuscation* of original data makes trivial content analysis unsuitable; in fact, even sophisticated techniques like Optical Character Recognition (OCR) can fail to recognize text in images, a fact that is often used in designing CAPTCHAs. This line of spam generation renders early i-spam detection techniques [6] ineffective.

- *I-spam messages use HTML effectively*: I-spam uses MIME to transport attached image data along with HTML formatting and non-suspicious text. However, the text contained in I-spam images and the text contained in the HTML body usually have no correlation, a property which can be exploited to classify i-spam. Regular text is usually taken from books or standard documents used to fool text-based email classifiers. Thus, the HTML text offers almost no help to test classifiers, unless appearing *too harmless* can be exploited as a feature.

- *I-spam messages differ from natural images*: Natural images tend to have smoother distribution in RGB/LAB color-space than I-spam. This property leads to better approximation of natural images by artificial distributions (e.g. Gaussian) than i-spam which often includes clear and sharp objects. This property is difficult to exploit in practice; nevertheless it is a significant property since ham emails with attached images contain very often natural images (e.g. photographs).

- *I-spam messages are template based*: A large number of I-spam are near-duplicates, i.e. a base pattern is permuted to form large number of similar looking but distinct I-spam images. Spammers exploit an existing pattern for a certain time period before investing in a new I-spam pattern. Such patterns are common in the world of text email spam, and are normally tackled by collaborative feedback from the user community. We envision the application of similar methods for new kinds audio-visual spam in systems like YouTube.

Available figures for 2006 suggest more than 30% of all spam emails are image based, and most of these emails reach email inboxes undetected. Humans are easily able to distinguish spam images from normal spam; moreover, they can abstract from various versions of the same base image, even though these images may have very different properties like background, fonts, or different arrangement of sub-images. Current feature based approaches have had some success in identifying image spam [5]; extracted features include file type and size, average color, color saturation, edge detection, and random pixel test. However these features do not seem to provide good generalization, since spammers can easily modify these features; thus further feature engineering is required. In particular, we observe that computer-generated images do not have the same texture as natural images; natural images like photographs would likely be classified as *ham*. We therefore postulate that low-level features which capture how an image is perceived are likely to be better discriminants between spam and ham. This is the first part of our detection strategy: to find features which capture how humans perceive spam images vs. ham images.

While some extracted features may provide good generalization, a spammer can clearly realize this and mutate spam images by using more natural/photographic content in spam images. However, creating completely different image spam each time is a costly activity; thus any spam image is likely to have thousands of similar (though not same) images, each generated from the same template image, but modified using the usual spam tricks. Even though image spam may evolve to defeat obvious give-away features (like text), there are likely to be many near duplicates. This forms the second part of our spam detection strategy: to detect near-duplicates of spam images which have been labeled as spam.

2.1 Low Level Features of Image Spam

As noted earlier, recent work has used feature-based classification to detect I-spam. Dredze et al. [5] have investigated the use of high-level features in classifying I-spam. Table 1 lists the features used by the authors in this work; the core idea is that high level features of the image, for example file format, can be extracted much quicker than lower level features like edges; thus there is an intrinsic ordering of features we want to consider in order to build fast classifiers. The approach is general and our work can easily be extended to this framework.

Other researchers (cf. [16]) have discussed yet another important aspect which we capture in our methodology: they have investigated common obfuscation methods like rotation, scaling, partial translation, font change etc. They correctly identify the procedure used by spammers to generate image spam; they postulate that the two steps involved are *template generation* and the *randomization of template*. This procedure makes images sent to different users different in nature, hence defeating simple techniques for checking exact duplicates (e.g. hashing). However, their method is limited in the features used, achieving an accuracy below 85%. In this work, we explore visual image features, while using a simple and novel mechanism of dealing with various obfuscation techniques. Table 2 lists obfuscation techniques that the survey in [16] has identified.

2.2 Other Methods for I-spam filtering

Early methods for I-spam exploited the heavy use of text in I-spam, a feature that is still present. However, spammers hit back by exploiting the deficiencies of Optical Character Recognition and modifying I-spam accordingly. Shortcomings of OCR have been aptly demonstrated in the design of CAPTCHAs[1] which remain an effective mechanism of telling humans and computer agents apart. To demonstrate the ineffectiveness of OCR for (current) I-spam filtering, we have compared our algorithm with OCR.

[1]www.captcha.net

Feature Type	No. Of Features
Average Color	44
Color Saturation	10
Edge Detection	32
File Format	332
File Size	6
Image Metadata	7195
Image Size	18
Prevalent Color Coverage	26
Random Pixel Test	11

Table 1: Features used for I-spam classification by Dredze et al. [5]

Method	Description
wave	using wavy text to fool OCR
animate	using animated frame
deform	deforming text using irregular fonts
rotate	rotate text to defeat OCR
shift	move template image
crop	cropping template image randomly
size	same image on different canvas size
dots	adding random pixels
bars	adding random lines in the image
frame	add a frame to the
font	use different font or size
line color	adding lines of random color
shape	use random shapes in background
metadata	randomize metadata
url	use different urls in image

Table 2: Randomization Methods used for I-spam generation by Wang et al. [16]

3. STRATEGIES FOR I-SPAM DETECTION

3.1 Near-duplicate Detection in Images

While technology may not be advanced enough for recognizing spam from random images without any explicit instructions or rules, recent research in machine learning has lead to efficient and accurate pattern recognition algorithms. Our near duplicate detection algorithm is based on the intuition that we can recognize a lot of images similar to an identified spam image; since I-spam is usually generated from a template (see Sec. 2), near duplicates should be easy to detect. Given enough training data, we should be able to detect large volumes of I-spam, while being open to further training. We should additionally provide better generalization performance than pure similarity search and abstract from observer positive and negative samples.

Near-duplicate detection for images is an extreme form of similarity search [7] which is a well studied topic. Recent work in probabilistic image clustering has focused on similar issues but from a different perspective, where a query image is used to search for similar images in a large database. Probabilistic image modeling is particularly suited to the task at hand since we want to classify a *family* of images as spam, while having observed only a few samples from the family. We have chosen Gaussian Mixture Models (GMM) as the starting point for our approach.

Gaussian Mixture Models model an image as coherent regions in feature space. First, features are extracted for each pixel, projecting the image to a high-dimensional feature space. For each pixel, we extract a seven tuple feature vector: two parameters for pixel coordinates (x, y) (to include spatial information), three for color attributes in the color space and two for texture attributes (*anisotropy* and *contrast* [3]). We chose the (L^*, a^*, b^*) color space (henceforth called as *Lab*) since it models most closely how humans perceive colors. The Lab color space is perceptually linear, meaning that a change of the same amount in a color value produces a change of about the same visual importance. For texture, anisotropy and contrast are extracted from the second moment matrix defined as follows:

$$M_\sigma = G_\sigma(x,y) * (\nabla I)(\nabla I)^T, \tag{1}$$

where $G_\sigma(x,y)$ is a separable binomial approximation to a Gaussian smoothing kernel with variance σ^2, and (∇I) is the gradient of the L channel in the *Lab* color-space. At every pixel, M_σ is a 2×2 matrix, and can be eigen-decomposed very simply. Considering a fixed scale σ, we compute the eigenvalues $\{\lambda_1, \lambda_2\}$ for M_σ at (x, y). Anisotropy are consequent defined as $A = 1 - \lambda_2/\lambda_1$ and contrast $c = 2\sqrt{\lambda_1 + \lambda_2}$.

Given this transformation, we now represent an image as a set of 7-tuples:

$$I = \{p_1, p_2, \cdots, p_n\},\ \ p_i = (x_i, y_i, L_i, a_i, b_i, A_i, C_i)\,, \tag{2}$$

where n is the number of pixel in the image. Given this representation with parameters , we model the probability distribution of I with a mixture model with K gaussians,

$$f(p|\Theta) = \sum_{j=0}^{K} \alpha_j f_j(p|\theta_j),\ \ s.t.\ \sum_{j=0}^{k} \alpha_j = 1 \tag{3}$$

where p is a feature vector, representing each point of an Image, and $\Theta = (\alpha_1, \cdots, \alpha_K, \theta_1, \cdots, \theta_K)$, and each f_j is a multivariate Gaussian density function parameterized by θ_j i.e., (μ_j and Σ_j). The probability distribution function $f_j(p|\theta_j)$ can then be written as

$$f_j(p|\theta_j) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_j|}}\, exp\{-\frac{1}{2}(p - \mu_j){\Sigma_k}^{-1}(p - \mu_j)^T\}$$

Additional constraints on the parameters of the above model include $\alpha_j > 0$, and diagonal covariance Σ_j; this simplifies the model and avoids inversions of the covariance matrix. We are now interested in finding the maximum likelihood estimate (MLE) of the model parameters Θ, such that:

$$\Theta_{ML} = \underset{\Theta^*}{\operatorname{argmax}}\, f(p_1, p_2, \cdots, p_n|\Theta^*) \tag{4}$$

Since no closed form solutions exist for the above maximization, Expectation Maximization [4] is used to find the MLE. The EM procedure alternates between E-steps and M-steps where the parameters are updated in the direction of maximum gain in log-likelihood. The EM equations for GMM are presented below:
E-step:

$$w_{ij} = \frac{\alpha_j f(p_i|\mu_j, \Sigma_j)}{\sum_{l=1}^{k} \alpha_l f(p_i|\mu_l, \Sigma_l)} \tag{5}$$

M-step:

$$\hat{\alpha_j} \leftarrow \frac{1}{n}\sum_{i=1}^{n} w_{ij} \tag{6}$$

$$\hat{\mu_j} \leftarrow \frac{\sum_{i=1}^{n} w_{ij} p_i}{\sum_{i=1}^{n} w_{ij}} \tag{7}$$

$$\hat{\Sigma_j} \leftarrow \frac{\sum_{i=1}^{n} w_{ij}(p_i - \hat{\mu_j})(p_i - \hat{\mu_j})^T}{\sum_{i=1}^{n} w_{ij}} \tag{8}$$

The GMM framework allows us to learn probabilistic models of images, but a hidden assumption is that the images are similar in size. While this is not true for our data, it offers an opportunity as well: *at lower resolutions, many obfuscation techniques are ineffective*. Using this key assumption, we scale every image to a resolution of 100×100. The computational advantage of this method is obvious. This scale has been selected empirically and strikes a balance between loss of information and gain in performance. We explore the effect of scale when extracting visual image features in Section 4.1.

In order to ensure faster GMM fitting, we optimized model fitting procedure; the EM model was parameterized using K-means clustering. For all experiments, the number of mixtures was fixed at $K = 4$. A smaller value of K would have resulted in omission of important components of an image, while a larger value would have taken into consideration also the insignificant portions of an image. After sufficient experimentation, the value of K was fixed at 4 to ensure that noise was filtered out of the model.

After all the GMMs have been learnt (one for every image), we need to cluster similar images together. Clustering requires a similarity measure, and our task requires modeling distance between two *probability distributions*. A commonly used distance measure is Kullback-Leiber (KL) Divergence defined as follows:

$$D_{\mathrm{KL}}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} \tag{9}$$

Even though it gives a measure of closeness of two distributions to each other, this measure is not symmetric. We therefore choose Jensen-Shannon divergence as the distance measure. The distance measure between clusters c_1 and c_2 takes into account both the dissimilarity between the probability distributions and the size of the two clusters as shown by the equation

$$D_{JS}(P||Q) = \frac{1}{2}(D_{\mathrm{KL}}(P||M) + D_{\mathrm{KL}}(Q||M)) \tag{10}$$

where $M = \frac{1}{2}(P + Q)$. Using this distance, we can perform clustering (Labeled Agglomerative Clustering); the idea is to find images which are similar enough in the training set, and to replace them with one signature. New images from the test set can now be compared to the already identified signatures and if there is a close match, then a positive identification can be made.

3.1.1 Labeled Agglomerative Clustering

In [8], the authors have described an agglomerative clustering algorithm, where clusters are learnt from observed data in an unsupervised fashion, i.e. the number of clusters is initially unknown. To aid in clustering, the *Information Bottleneck* principle is used. The information bottleneck

(IB) principle states that the loss of mutual information between the objects and the features extracted should be minimized by the desired clustering among all possible clusterings. Using the IB principle, clustering of the object space X is done by preserving the relevant information about another space Y . We assume, as part of the IB approach, that $\hat{X} \rightarrow X \rightarrow Y$ is a Markov chain, i.e. given X the clustering $\hat{X}$ is independent of the feature space Y . Consider the following distortion function:

$$d(x, \hat{x}) = D_{\text{KL}}(p(y|X = x)||p(y|\hat{X} = \hat{x})) \qquad (11)$$

where D_{KL} is as defined above in (10).

Note that $p(y|\hat{x}) = \sum_x p(x|\hat{x})p(y|x)$ is a function of $p(\hat{x}|x)$. Hence, $d(x|\hat{x})$ is not predetermined, but depends on the clustering. Therefore, as we search for the best clustering, we also search for the most suitable distance measure.

Since the number of clusters may not be known *apriori*, we use the agglomerative clustering. The Agglomerative Information Bottleneck (AIB) algorithm for clustering is a greedy algorithm based on a bottom-up merging procedure [13] which exploits the IB principle. The algorithm starts with the trivial clustering where each cluster consists of a single point. Since we want to minimize the overall information loss caused by the clustering, in every greedy step we merge the two classes such that the loss in the mutual information caused by merging them is minimized.

Note however, that we are still operating in the classification domain, i.e. the data available to us has a label (*ham/spam*). The original AIB approach does not consider data labels as input; therefore, we extend it here for handling label data. We can use label information to restrict the clustering to a similar data, thus speeding up the cluster formation by distance comparison with data of the same label. Since the base AIB algorithm is quadratic, this information can result in a speedup of up to 4 times (assuming labels are equally likely to be positive and negative). Further optimizations are possible to make the unsupervised clustering faster (e.g. merging more than two clusters in an iteration or merging below a threshold); this is a placeholder for replacement with better Labeled AIB algorithms. Our presented algorithm consists of the two phases GMM training and Labeled AIB clustering which are summarized in Algorithms 1 and 2.

Algorithm 1 TrainGMM $(I_i, i = \{1..N\}, k)$

Require: Images $I_i, i = \{1..N\}$, No of Mixtures k

```
 1: for  $i = 1$ to $N$ do
 2:     Resize $I_i$ to $100 \times 100$ pixels.
 3:     Extract (x,y,L,a,b,A,c) features for each pixel.
 4:     Perform K-means clustering on $I_i$.
 5:     Choose $\alpha_j$ randomly, s.t on $\sum_j^k \alpha_j = 1$.
 6:     Initialize $\mu_j$ to center of cluster $j$.
 7:     Initialize $\Sigma_j$ to $\mathbf{1}$.
 8:     while Convergence not reached do
 9:         Repeat E-steps and M-steps (Eq. (5)-(8))
10:     end while
11: end for
```

Ensure: $\Theta = (\mu_j,\ \alpha_j,\ \Sigma_j)$

Algorithm 2 LabeledAIB (θ_i, L_i)

Require: GMMs $\theta_i, i = \{1..N\}$, Label L_i for each image

```
 1: $C = \{c_1, \cdots c_N\}$ {$N$ clusters }
 2: for  $i = 1$ to $N$ do
 3:     $f(p|c_i) = \{\theta_i\}$ {Each image in its own cluster}
 4: end for
 5: for each $i, j$ pair, $i, j = 1$ to $N$ do
 6:     Distance$(i, j)$ = JS$_{dist}(f(p|c_i), f(p|c_j))$
 7: end for
 8: for $i = 1$ to $N - 1$ do
 9:     repeat
10:         Find $\{m, n\} = \underset{m,n}{\text{argmin}}\, |\text{Distance}(i, j)|$
11:     until  $L_m == L_n$, else find next minimum
12:     if  Distance$(m, n) \leq Threshold$ then
13:         Break
14:     end if
15:     Merge $\{c_m, c_n\} = \hat{c}$.
16:     $f(\hat{c}) = \frac{|c_m|}{|c_m \cup c_n|} f(c_m) + \frac{|c_n|}{|c_m \cup c_n|} f(c_n)$
17:     Update Distance$(i, j)$
18: end for
```

Ensure: $f(p|c_l)$, c_l, $l \leftarrow$ No. Of Clusters

3.1.2 Prediction Phase

New images, whose label have to be predicted, are fitted to a GMM, and compared to the labeled signatures already known from the trained cluster model. The closest cluster to a new image, is found using the JS divergence distance, and the label of the new image is the label of the cluster. As a result, images similar to the signatures corresponding to spam label will be marked as spam and images similar to ham labeled signatures will be marked as ham. An optimization of this algorithm is to introduce a new label called *unidentified*. Images that are not similar to any of the existing signatures (i.e. the JS divergence with the closest cluster is larger than a particular threshold) can be labeled as unidentified and may require user intervention to be labeled appropriately into either of ham or spam data sets. This way more accurate results are obtained while also ensuring that the training phase continues even in the testing phase.

3.2 Visual Features for Classification

Near-Duplication is likely to perform well in abstracting base templates, when given enough examples of various spam templates in use. However, the generalization ability of this method will be limited, since we are not exploiting global features of images; there are many likely *giveaway* features as previous work has shown. We feel however that the explored feature set does not identify a key aspect: *i-spam is artificially generated.* In this section, we explore visual features like texture, shape and color and learn classifiers using these selected features. Notice that we extract global features, i.e. each feature represents a property of the entire spam image.

3.2.1 Color Features

Color models are used to classify colors and to qualify them according to such attributes as hue, saturation, chroma, lightness, or brightness. The *Red-Blue-Green*

(RGB) model is the most basic and well-known color model. It is also the basic color model for on-screen display. Using this color model, we chose the following features

- *Average RGB*: Average RGB represents the average values in R, G and B channel of each pixels in an image. The range of RGB is normalized to $(0, 1)$.
- *Color Histogram*: For a given image, the color histogram H_I is a compact summary of the image. A color histogram $\mathbf{H}$ is a vector $(h_1, h_2, \cdots, h_n)$, in which each bucket h_j contains the number of pixels of color j in the image. Typically images are represented in the RGB color-space, and a few of the most significant bits are used from each color channel. We chose a 6-bit color space leading to 64 feature vectors.
- *Color Moment*: The use of color moments is based on the assumption that the distribution of color in an image can be interpreted as a probability distribution. Probability distributions are characterized by a number of unique moments; [14] uses three central moments of a image's color distribution; they are *mean, standard deviation* and *skewness*. Using RGB channels and 3 moments for each channel, we get 9 feature vectors.
- *Color Coherence Vector*: An image's color coherence is the degree to which pixels of that color are members of large similarly-colored regions. We refer to these significant regions as coherent regions and aim to classify pixels as coherent or incoherent. We first blur the image slightly by replacing pixel values with the average value in a small local neighborhood (currently including the 8 adjacent pixels). This eliminates small variations between neighboring pixels. We then discretize the color space, such that there are only n distinct colors in the image. The next step is to classify the pixels within a given color bucket as either coherent or incoherent. A coherent pixel is part of a large group of pixels of the same color, while an incoherent pixel is not. We determine the pixel groups by computing connected components. Note that we only compute connected components within a given discretized color bucket. This effectively segments the image based on the discretized color-space. We classify pixels as either coherent or incoherent depending on the size in pixels of its connected component(C). A pixel is coherent if the size of its connected component exceeds a fixed value T (e.g. 1% of number of pixels); otherwise, the pixel is incoherent. 128 feature vectors are chosen in this manner.

3.2.2 *Texture features*

Image texture, defined as a function of the spatial variation in pixel intensities (gray values), is useful in a variety of applications and has been a subject of intense study by many researchers (cf. [15]). The intuition behind choosing texture features for classification is that natural images have a different quality of texture as compared to textures in computer generated images. We extract the following features from the images in our data set:

- *Autocorrelation*: Autocorrelation measures the coarseness of an image by evaluating the linear spatial relationships between texture primitives. Large primitives give rise to coarse texture (e.g. rock surface) and small primitives give rise to fine texture (e.g. silk surface). If the primitives are large, the autocorrelation function decreases slowly with increasing distance whereas it decreases rapidly if texture consists of small primitives. If the primitives are periodic, the autocorrelation function increases and decreases periodically with distance.

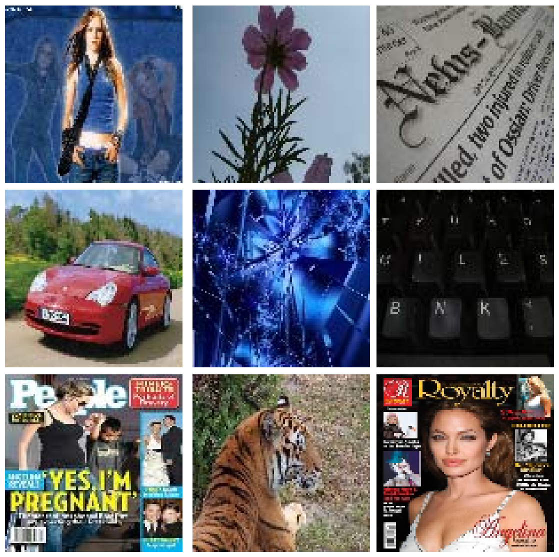


Figure 2: Examples of Ham images chosen by us for training the classifier. Note that we chose some ham images with text as well, other categories of ham images used company logos, cartoons and wall papers.

- *Edge Frequency*: A number of edge detectors can be used to yield an edge image from an original image. We can compute an edge dependent texture description function E as follows:

$$\begin{aligned} E =& |f(i,j) - f(i,j+d)| + |f(i,j) - f(i,j-d)| \\ &+ |f(i,j) - f(i+d,j)| + |f(i,j) - f(i-d,j)| \end{aligned}$$

  This function is inversely related to the autocorrelation function. Texture features can be evaluated by choosing specified distances d and averaging over the entire image. We vary the distance d parameter from 1 to 25 giving us a total of 25 features.
- *Primitive Length (Run Length)*: A primitive is a continuous set of maximum number of pixels in the same direction that have the same gray level. Each primitive is defined by its gray level, length and direction. Primitive length (run length) uses lengths of texture primitives in different directions as texture description. Coarse textures contain more long texture primitives, and fine textures contain more short texture primitives.
- *Co-occurrence Matrices*: Whether considering the intensity or gray-scale values of the image or various dimensions of color, the co-occurrence matrix can

measure the texture of the image. Because co-occurrence matrices are typically large and sparse, often various metrics of the matrix are taken to get a more useful set of features. Features generated using this technique are usually called *Haralick* features [9]. Co-occurrence Matrices are based on repeated occurrence of some gray-level configuration in the texture; this configuration varies rapidly in fine textures, though more slowly in coarse textures.

3.2.3 Shape features

- *Geometric Moment*: Image moments are certain weighted averages (moments) of the image pixels' intensities, or functions of those moments, usually chosen to have some attractive property or interpretation. When normalized, they can be considered as a probability distribution. If $f(x, y)$ is a digital image, the central moment is defined as follows:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x, y) \qquad (12)$$

The central moment is useful in representing the orientation of the image content.

- *Eccentricity*: An approximate measure of eccentricity or elongation of an object is given by

$$e = \frac{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}\mu_{11}}{(\mu_{20} + \mu_{02})^2} \qquad (13)$$

Where μ_{ij} is the i, j^{th} moment as defined in Eq. (12).

- *Legendre and Zernike Momemts*: Global moments are employed as the invariant global features of an image in pattern recognition due to their ability to recognize shapes. The use of non-geometric moments has been advocated in image reconstruction due to their better resistance to noise and more accurate reconstruction. Legendre moments use Legendre polynomials while Zernike moments use Zernike polynomials.

3.3 An algorithm for Classification using Visual Features

Many approaches exist to train classifiers using extracted features; however, *Support Vector Machines*(SVM) [2] have been established as the best performing technique. In particular, the use of the *kernel trick* allows SVM to explore non-linear combinations of features by considering dot products in a high dimensional Hilbert space. The SVM classification is formalized as follows: We consider data points of the form:

$$\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \ldots, (\mathbf{x}_n, c_n)\} \qquad (14)$$

where $c_i \in \{-1, 1\}$ and denotes the class that $\mathbf{x}_i$ belongs to (e.g. spam/ham).We can view this as training data, which denotes the correct classification which we would like the SVM to distinguish, by means of the dividing (or separating) hyperplane, which takes the form

$$\mathbf{w}.\mathbf{x} - b = 1, \ |\mathbf{w}| = 1 \qquad (15)$$

The kernel trick is used in the dual form of quadratic problem solving the above optimization. The SVM$_{\text{light}}$ package [10] offers an efficient implementation of SVMs with many supported kernels. Since data might not be separable, soft margin classification may be required. We have used the radial basis function as a kernel function since the corresponding Hilbert space is of infinite dimension. Algorithm 3 summarizes the overall approach.

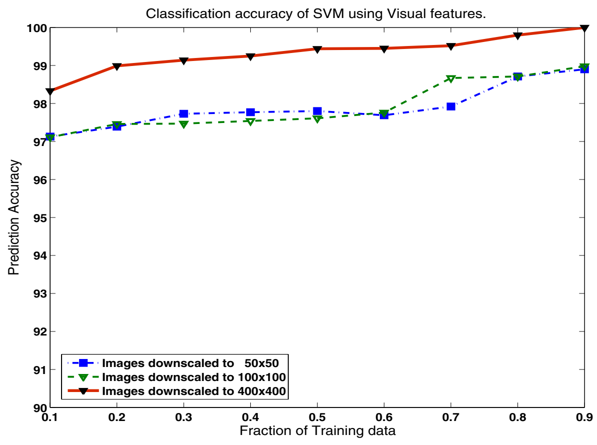


Figure 3: Classification Accuracy of Visual Features based-SVM compared at different resolutions. Notice how higher resolutions provide better results, but at much higher costs, since most texture and shape features have non linear extraction time.

Algorithm 3 SVM-classify (I_i, L_i)

Require: Images I_i, Label $L_i, i = \{1..N\}$,

```
1: for $i = 1$ to $N$ do
2:     $\mathbf{F}_i^{color} = ExtractColorFeatures(I_i)$
3:     $\mathbf{F}_i^{shape} = ExtractShapeFeatures(I_i)$
4:     $\mathbf{F}_i^{texture} = ExtractTextureFeatures(I_i)$
5:     $\mathbf{F}_i = \mathbf{F}_i^{color} \cup \mathbf{F}_i^{shape} \cup \mathbf{F}_i^{texture}$
6: end for
7: Classifier C = $LearnSVMClassifier(\mathbf{F}, \mathbf{L})$
```

Ensure: Classifier C

3.4 Optical Character Recognition

Optical Character Recognition (OCR) was the first proposed solution to I-spam; there are various commercial and open source solutions (e.g. SpamAssasin's FuzzyOCR plug-in) using OCR libraries. We chose to use the well known *Tesseract* OCR suite developed by HP labs, recently open sourced by Google[2]. The OCR based algorithm we use is a simple one: we classify an image as spam if the OCR module can find more than 2 characters in the image. This has low accuracy of around 80%; increasing the threshold to 5 characters leads to accuracies below 65%.

4. EVALUATION

To evaluate our algorithms, we have chosen three recent public datasets; the first is due to [6] who collected over

[2]Tesseract is available at `http://code.google.com/p/tesseract-ocr/`

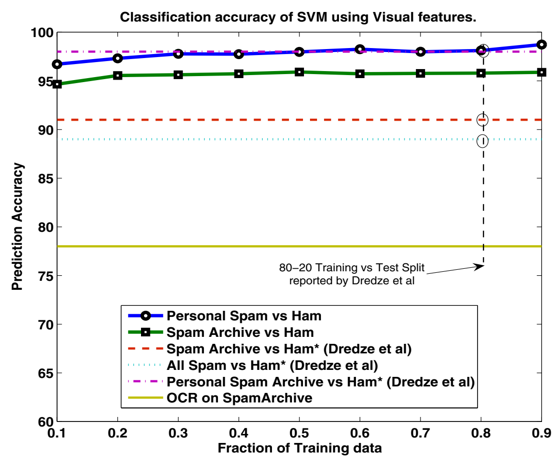


Figure 4: Classification Accuracy of Visual Features based-SVM compared with previous results. Notice a significant improvement over other approaches and good generalization results at training sizes as low as 10%.

13000 I-spam emails from the erstwhile SpamArchive[3]. The second was created by Dredze et al. [5] (who also used the SpamArchive dataset for evaluation) from their personal emails. This is called the *Personal Spam* dataset by the authors; they have also created a *Personal Ham* dataset from their personal emails, representing true ham emails. This is an important collection since most other researchers have used general images and photos from the web (examples of ham images are shown in Fig. 3.2.1). Both these datasets are now publicly distributed at `http://www.seas.upenn.edu/~mdredze/datasets/image_spam/`. The SpamArchive Collection consists of 13621 files, out of which only 10623 are in image format. The other files are unreadable by image processors; this is due to deliberate manipulation by spammers to use other alternatives of image for e-mail spam. The Personal spam collection consists of 3300 images. In addition, we have also used the Princeton Image Spam Benchmark[4] which contains 1071 images with category information. The utility of this dataset is that each category contains permutations of the same base I-spam template (See Fig. 6).

Baseline: For the sake of comparison, we use the same datasets used previously by other researchers. In particular, we use the approach for testing as used by Dredze et al. since their results are the best reported so far in comparison to other work. The above presented algorithms are compared in identical conditions as in the experiments in [5]. OCR results using Open Source Tesseract are also provided for comparison.

4.1 Results

We first discuss the more general approach of binary classification using visual features. We use our homegrown ham data set consisting of images, photographs, logos, wallpapers, emoticons and similar items used in real world email exchanges. In all, this amounts to 5373 ham images created from collected images including a large chunk of Dredze et al.'s personal ham (which has not been released in its entirety due to privacy reasons). We first resize the images to 100 × 100 pixels; this simple mechanism makes our method robust to random pixels, simple translation and scaling. It also helps greatly with computational requirements as texture and shape extraction are time consuming non-linear routines. We then extract features from all the images from the positive and negative test set. A fraction α of the images are used for training the classifier based on SVM^{light} while the tests are then carried out on the remaining (1-α) fraction of images. Fig. 4 reports the prediction accuracy over the positive and negative set as function of α.

We observe that visual features are highly indicative of spam images; our method reports a prediction accuracy of over 95% in all cases. Even with large sets of over 15,000 emails with only 10% data used for training, the prediction accuracy is higher than best numbers reported by [5, 6]. At a comparable fraction of training/test set as used by Dredze et al., we achieved almost 98% accuracy, an improvement of over 6%. With the Personal spam dataset, we achieve comparable results as [5].

We also investigated the impact of resolution on our approach; in particular we are interested to know if lower resolutions can provide similar results at a cheaper computationally cost. Fig. 3 shows the results of this evaluation; we notice small losses at lower resolutions and small gains at higher resolutions. At 400 × 400, the approach is near

[3]The site was earlier available at `http://www.spamarchive.org/`.
[4]`http://www.cs.princeton.edu/cass/spam/spam_bench/`

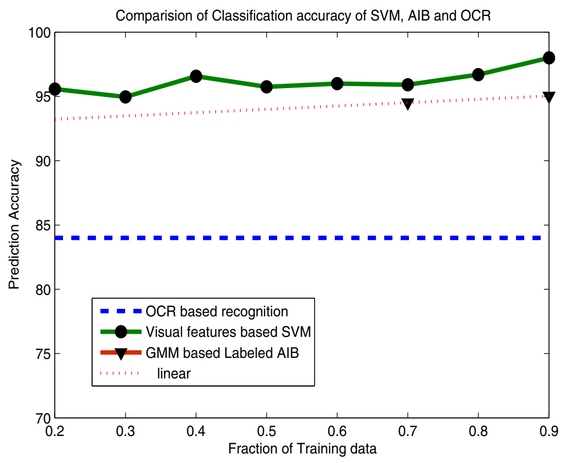
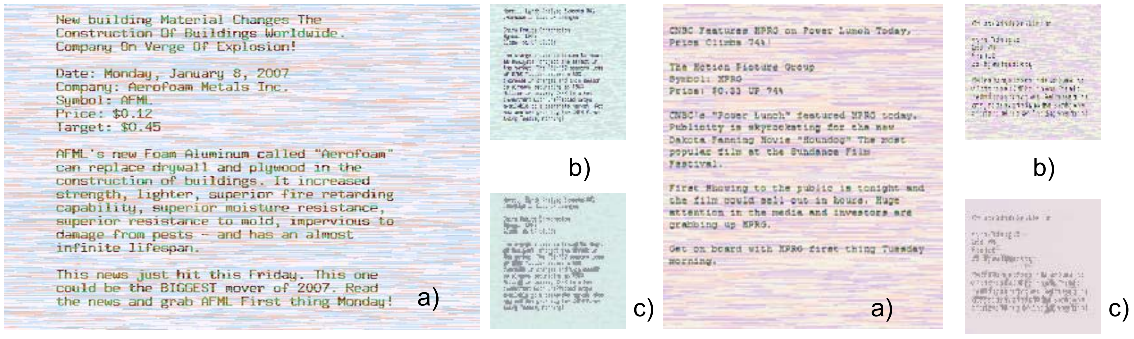


Figure 5: Comparison of Classification Accuracy of Visual Features based-SVM along with GMM based AIB and OCR (Princeton Dataset).

GMM Based AIB	
# Original Clusters	178
# Clusters Found	140
# Total Images	1004
# Images Correctly classified	760
# Images Misclassified	244
# % Error	24%

Table 3: Clustering Accuracy of GMM based Labeled AIB on the Princeton Spam Benchmark


New building Material Changes The
Construction Of Buildings Worldwide.
Company On Verge Of Explosion!

Date: Monday, January 8, 2007
Company: Aerofoam Metals Inc.
Symbol: AFML
Price: $0.12
Target: $0.45

AFML's new Foam Aluminum called "Aerofoam"
can replace drywall and plywood in the
construction of buildings. It increased
strength, lighter, superior fire retarding
capability, superior moisture resistance,
superior resistance to mold, impervious to
damage from pests - and has an almost
infinite lifespan.

This news just hit this Friday. This one
could be the BIGGEST mover of 2007. Read
the news and grab AFML First thing Monday!

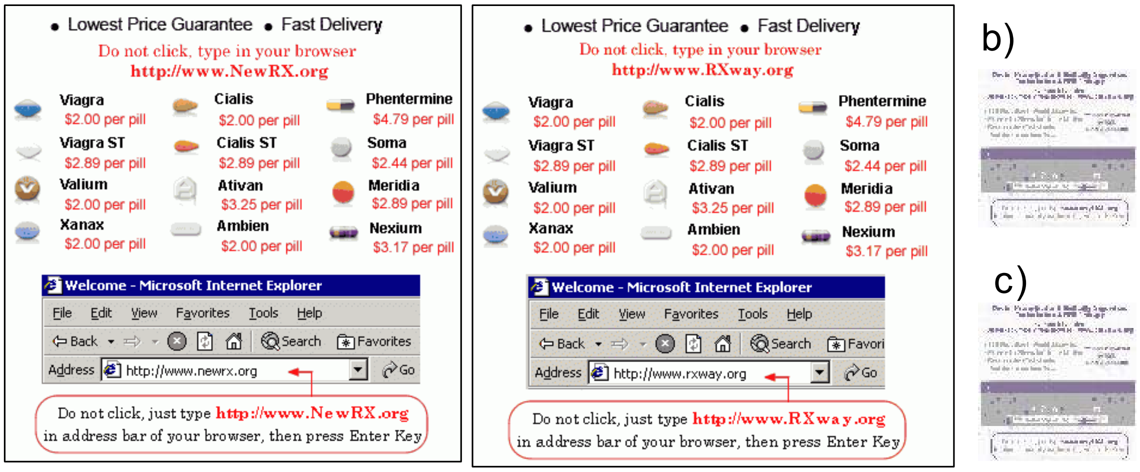
CNBC Features MPRG on Power Lunch Today,
Price Climbs 74%!

The Motion Picture Group
Symbol: MPRG
Price: $0.33 UP 74%

CNBC's "Power Lunch" featured MPRG today.
Publicity is skyrocketing for the new
Dakota Fanning Movie "Houndog" The most
popular film at the Sundance Film
Festival.

First Showing to the public is tonight and
the film could sell out in hours. Huge
attention in the media and investors are
grabbing up MPRG.

Get on board with MPRG first thing Tuesday
morning.


• Lowest Price Guarantee • Fast Delivery
Do not click, type in your browser
http://www.NewRX.org

Viagra $2.00 per pill	Cialis $2.00 per pill	Phentermine $4.79 per pill
Viagra ST $2.89 per pill	Cialis ST $2.89 per pill	Soma $2.44 per pill
Valium $2.00 per pill	Ativan $3.25 per pill	Meridia $2.89 per pill
Xanax $2.00 per pill	Ambien $2.00 per pill	Nexium $3.17 per pill

Do not click, just type http://www.NewRX.org in address bar of your browser, then press Enter Key

• Lowest Price Guarantee • Fast Delivery
Do not click, type in your browser
http://www.RXway.org

Viagra $2.00 per pill	Cialis $2.00 per pill	Phentermine $4.79 per pill
Viagra ST $2.89 per pill	Cialis ST $2.89 per pill	Soma $2.44 per pill
Valium $2.00 per pill	Ativan $3.25 per pill	Meridia $2.89 per pill
Xanax $2.00 per pill	Ambien $2.00 per pill	Nexium $3.17 per pill

Do not click, just type http://www.RXway.org in address bar of your browser, then press Enter Key

Welcome - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.newrx.org Go
Welcome - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favori
Address http://www.rxway.org Go
b)
c)

Figure 6: Examples of two I-spam images from the same category in the Princeton dataset. In the figure on top, (b) is the 100X100 image, and (c) is the reproduced image. In the figure below, the small figures are GMM reconstructions of 100X100 thumbnails. Notice how close the reproduction of the low-res image is to the actual low-res image. Even though the spam images are different in resolution and content (see the figure below), the low res reproduction are almost identical to the GMM approximation.

perfect achieving more than 99.6%. Higher accuracies will be practically impossible due to human errors in labeling data.

It is important to point out that our results are nearly completely comparable with the previous papers; all researchers have used their own ham datasets, including us. The choice of ham can have a big impact on the outcome of prediction. However, the important issue is labeling spam correctly over a large heterogeneous spam set. We are currently procuring more spam by creating *honeypots* with an aim of collecting over 100,000 spam emails. We are also working on the creation of a *SpamAssassin*[5] Plug-in using our framework.

Results from GMM based AIB The Labeled AIB approach is unlikely to reach the same generalization performance; the approach is designed to identify permutations of a base template. Since positive images do not necessary follow this pattern and are potentially infinite, Labeled AIB is expected to be clueless about images previously unseen. Early experiments in settings as above lead to weak results. However, the idea is still powerful; collaborative approaches to spam filtering like *Spamato* [1] encourage users to share manually-classified email spam, which can be used to train the spam filters of other users in the community. Our approach fits very well into this framework, since AIB based on GMMs can identify the base template. This makes our approach also well suited to server deployment as an identified pattern can be protected against for a large user population before spam reaches individual inboxes. The goal is to detect a new pattern in a manner similar to viruses and make the pattern available to all subscribed email servers. Further investigation of this idea is in progress.

To explore how effective the GMM approach is in identifying patterns pre-classified as spam, we find the Princeton dataset very helpful. This dataset has various categories of I-spam images clustered according to the base template; descriptions of different strategies used by spammers are also provided. Fig. 6 demonstrates that two spam images produced by the same template are indeed modeled very similarly to each other. Notice how well our chosen features can model the low resolution *thumbnails* with a simple parametric model. This provides the perfect opportunity to raise the question: *Can AIB Clustering based on GMM identify the clusters correctly and demonstrate that it recognizes the base templates?* We performed a run with our algorithm to test this hypothesis.

The Princeton dataset contains images in 178 categories,

[5]`www.spamassasin.org`

however this categorization is very strict. Many of these clusters can be merged and manual clustering resulted in fewer than 150 clusters. Our approach found 140 clusters; clearly some misclassification was also found. However, the performance is better than indicated by the numbers; clusters that were wrongly merged were classified as wrong, though they are similar when manually observed. When constrained to find exactly 178 clusters, the misclassification rate falls to below 16%. This indicates that the predictive performance for spam detection should still be high. Fig. 5 proves this empirically. GMM based Labeled AIB has high accuracy when predicting spam in comparable conditions. OCR does much worse, while the trained SVM has a marginally better performance.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented two novel approaches to fight the menace of image-based email spam by exploiting visual features and the template-based nature of I-spam. Experimental results heavily support our hypothesis that low-level feature analysis can provide a more accurate detection mechanism than previously known. Our work complements earlier work, and it should be very easy to incorporate visual features extracted by us into the framework of [5]. The gain is likely to be better accuracy, with a much improved running time by using the JIT feature extraction suggested by them. Since research in I-spam is recent, with less than one year since the emergence of the problem, more work is likely to happen in the future. In particular, spammers will notice the anti-spam measures taken and innovate to produce new attacks. The emergence of PDF based spam is one such innovation from spammers; clearly, spammers try to exploit all popular document formats to get their messages through. Till more principled shifts in email (e.g. postage for email c.a. [11]) or improvements in the underlying protocols happen, anti-spam research will remain a firefighting operation.

6. REFERENCES

[1] K. Albrecht, N. Burri, and R. Wattenhofer. Spamato-An Extendable Spam Filter System. *2nd Conference on Email and Anti-Spam (CEAS), Stanford University, Palo Alto, California, USA*, 2005.

[2] C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[3] C. Carson, M. Thomas, S. Belongie, J. Hellerstein, and J. Malik. Blobworld: A system for region-based image indexing and retrieval. *Third International Conference on Visual Information Systems*, pages 509–516, 1999.

[4] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[5] M. Dredze, R. Gevaryahu, and A. Elias-Bachrach. Learning Fast Classifiers for Image Spam. *In proceedings of the Conference on Email and Anti-Spam (CEAS), 2007*, pages 487–493, 2007.

[6] G. Fumera, I. Pillai, and F. Roli. Spam Filtering Based On The Analysis Of Text Information Embedded Into Images. *The Journal of Machine Learning Research*, 7:2699–2720, 2006.

[7] J. Goldberger, S. Gordon, and H. Greenspan. An efficient image similarity measure based on approximations of KL-divergence between two gaussian mixtures. *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 487–493, 2003.

[8] J. Goldberger, H. Greenspan, and S. Gordon. Unsupervised Image clustering using the Information Bottleneck method. *Proc. DAGM*, 2002.

[9] R. Haralick, I. Dinstein, and K. Shanmugam. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 3:610–621, 1973.

[10] T. Joachims. Making large-scale SVM Learning Practical. *Advances in Kernel Methods-Support Vector Learning.*

[11] R. Kraut, J. Morris, R. Telang, D. Filer, M. Cronin, and S. Sunder. Markets for attention: will postage for email help? *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 206–215, 2002.

[12] Secure Computing. Image spam: The latest attack on the enterprise inbox. Secure Computing Whitepaper, available online, Nov 2006.

[13] N. Slonim and N. Tishby. Agglomerative information bottleneck. *Advances in Neural Information Processing Systems*, 12:617–23, 2000.

[14] M. Stricker and M. Orengo. Similarity of color images. *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 2420:381–392, 1995.

[15] M. Tuceryan and A. Jain. Texture analysis. *Handbook of Pattern Recognition and Computer Vision*, pages 235–276, 1993.

[16] Z. Wang, W. Josephson, Q. Lv, M. Charikar, and K. Li. Filtering Image Spam with Near-Duplicate Detection. *Proceedings of the 4th Conference on Email and Anti-Spam (CEAS)*, 2007.