IDENTIFYING IMAGE SPAM AUTHORSHIP WITH VARIABLE BIN-WIDTH HISTOGRAM-BASED PROJECTIVE CLUSTERING

Song Gao, Chengcui Zhang, Wei-Bang Chen
The University of Alabama at Birmingham, Birmingham, AL 35294, USA
{gaos, zhang, wbc0522}@cis.uab.edu

ABSTRACT

In this paper we present a two-phase spam image clustering framework. The proposed framework performs a histogram-based projective clustering on visual features in the first phase, followed by a text-based clustering in the second phase. There are several contributions in this study. First, we address the complex nature of spam image obfuscation techniques. Second, a multi-clue framework is developed to profile spam images of common spamming sources which provide evidence for tracking spam gangs. Third, projective clustering eliminates the need to choose among distance metrics for clustering analysis, while systematically exploring subspaces that correspond to clusters.

Index Terms— Spam image clustering, histogram-based projective clustering, wavy spam image correction

1. INTRODUCTION

Spam e-mails are unsolicited bulk e-mails that impact our daily life by not only wasting our time but also tricking recipients into exposing sensitive information to cyber criminals. Although various mechanisms have been established to block spam e-mails, spammers have been using various tactics to elude these anti-spam techniques.

Image spam is known to be one of the commonly used approaches. Spammers can simply create a set of seemingly different images from a common template with some minor modifications. Commonly used obfuscation techniques include the use of different colors, varying the space between words and lines, randomly adding speckles, changing font size, splitting up one word into two halves with a gap in between, and repositioning or replacing embedded graphics. These minor changes make the images from a common spamming source visually similar to human eyes but essentially unique to fingerprinting techniques such as MD5 (Message-Digest algorithm 5) [2]. Clustering techniques can be applied to reveal common origins of spam images. We previously proposed several distance-based clustering methods in this research field [1, 15]. However, several problems remain unsolved.

First, these methods heavily rely on proper selection of distance (or similarity) metrics, However, distance functions may not faithfully describe how dissimilar objects are and can be highly application specific. Clustering results are also sensitive to the selection of the similarity threshold, such as [1]. In addition, distance measures usually require two objects to have the same set of features; however, not all of the features can be extracted from each object at all time. For example, some spam images have text content but others do not. Therefore, additional effort is needed to design measures to evaluate the goodness fit of a cluster with application specific thresholds. Moreover, an object may have categorical features which cannot be directly incorporated into a single distance metric. Therefore, applying distance-based clustering on spam image analysis may not be the best choice. In order to tackle these problems, we introduce a variable bin-width histogram-based projective clustering algorithm in this study.

Compared with distance-based clustering, projective clustering can detect clusters corresponding to feature subspaces. Moreover, a new variable bin-width histogram is proposed as a middle ground between kernel density estimation and fixed bin-width histograms by considering both efficiency and effectiveness.

Second, although optical character recognition (OCR) has been widely adopted to extract embedded text from spam images, it is known to be an error-prone process, which is even worse when applied to image spam due to the fact that the spam images are usually very noisy with a low resolution. In [1], we introduce the use of Needleman-Wunsch algorithm to improve the inaccuracy in extracting text. However, none of existing approaches can effectively extract texts from wavy images (the 1st and the 3rd images in Figure 1) which have recently been identified by Symantec as a spam campaign. The wavy images adopt a novel swirly image trick to defeat OCR since OCR cannot manage an image distorted to such a degree. In this paper, one particular contribution is the design of a robust algorithm to correct wavy image (the 2nd and the 4th images in Figure 1) so that OCR can be effectively applied.

Original Corrected Original Corrected

Fig. 1. Two examples of wavy images.

2. THE PROPOSED METHOD

A histogram-based clustering framework is proposed to reveal the origins of spam images through the following steps: (1) image preprocessing that includes wavy image correction and spam image segmentation, (2) features extraction, (3) two-phase clustering, including the proposed histogram-based projective clustering in the first phase, followed by the text-based clustering in the second phase.

978-1-61284-350-6/11/$26.00 ©2011 IEEE

2.1. Wavy image correction

Spammers create a set of varying wavy images by shifting each vertical line of pixels upward or downward associated with other obfuscation techniques. The shifting process makes horizontal lines 'wavy'. To extract the embedded texts from wavy images, correction needs to be done by realigning each vertical line to its correct position. Two perceivable approaches can be used to find the guideline based on which realignment can be done.

The first approach constructs a guideline based on a color-matching scheme. This approach finds the best color match of two adjacent vertical lines by fixing one line and slightly shifting the other line upward or downward. The naïve implementation first converts a wavy image into 6-bit color-code, and then, partitions the image into spatially connected segments (objects) based on the coded colors, followed by a color-matching process. This method is effective when applied to an artificially created wavy image which contains solely horizontal bands before image distortion. However, this approach suffers from severe noises and varying object shapes when applied to real wavy images, partly because it assumes that the upper and lower horizontal edges of any object must be parallel in the undistorted image. This assumption is not always true due to the fact that wavy images usually contain objects with irregular shapes, e.g., the pills and the triangle sign in Figure 1. Therefore, we further analyze the variance of height distribution of each object. A lower variance in height suggests that the upper and lower horizontal edges of an object are more likely to be parallel. Therefore, when aligning two adjacent vertical lines, we only perform color-matching scheme on the vertical line sections which belong to objects with low variance in height. Figure 2 illustrates the idea of the color-matching scheme. After matching two vertical lines n and n+1 based on line sections belonging to objects with low height variance only (e.g., the bottom object in Figure 2), an offset value can be obtained for line n+1 relative to line n. A set of such offset values can be used as a guideline to correct the wavy image.

The second approach finds the realigning guideline in a wavy image by identifying the curve(s) based on which the original image is distorted horizontally. In other words, these curve lines are originally horizontal lines in the undistorted image which could serve as a guideline for image correction. In order to obtain the edge map, the Laplacian of Gaussian (LoG) operator [3] is convoluted on a wavy image. Then, irrelevant edges are removed by using an edge direction histogram based method. The basic assumption is that if a continuous edge segment consists of significant portion of edges of opposite directions, it cannot be a horizontal line before it is distorted. For example, in Figure 1, the edges of texts and the pill object contours both fall into this category, while the upper and bottom edges of the window search bar have relatively simpler edge direction histogram. After filtering, all the remaining edges can be used to calculate the offset value for guideline construction. We start from the longest edge in the horizontal direction, and the offset value can be calculated as the vertical displacement between adjacent columns of pixels. This process iterates till either (1) a complete guideline that covers the full horizontal range is constructed without any gap, or (2) no more edges can be used to construct the guideline. According to our experiments, this method works well in finding most of the edges on a guideline. However, it is quite often to see gaps in between edge segments on a guideline (see Figure 3) due to imperfect edge detection. In other words, these edges cannot cover the entire horizontal range.

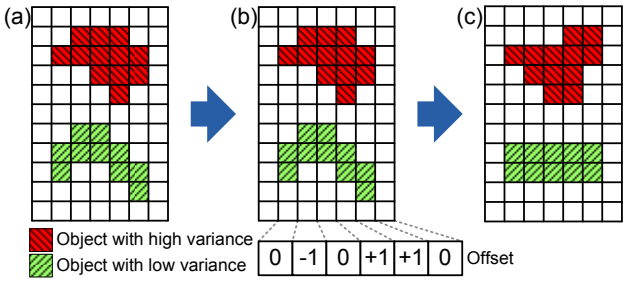

Fig. 2. The idea of color-matching approach. (a) A wavy image; (b) Calculating adjacent column offset; (c) The corrected image.

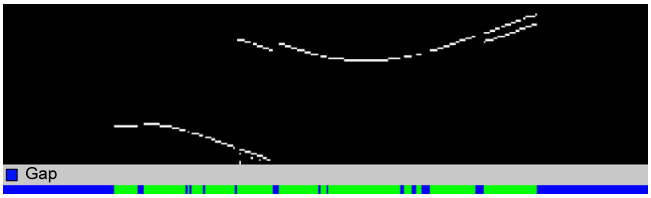

Fig. 3. Gaps between edge segments in the constructed guideline.

Our framework combines both approaches. The edge-based approach is first applied, and then, the color-matching scheme is used to cover gaps in the guideline. Two examples of corrected wavy images are shown in Figure 1.

2.2. Image segmentation

According to the image composition, spam images are usually composed of three components, including text content, foreground illustration, and background. Spammers often produce a set of visually similar images by slightly changing the content in one or more of these components. For example, substituting portion of text content or replacing part of foreground graphics. Thus, we may assume that two images of a common source must share some common traits in one or more of these components. In order to measure the similarity between corresponding components in two images, it is essential to perform object segmentation for component extraction. These components are extracted by applying the approach in [1] but with an improved background detection method. Instead of using 6-bit RGB color-code in [1], the proposed framework adopts a HSV histogram representation of an image since the HSV color space matches more perceptually the way that human perceive color. Using HSV histogram with 256 bins for each channel also avoids the information loss when

representing an image with 6-bit color-code. To this end, an image is segmented into the abovementioned three components (see Figure 4). Then visual and text features can be extracted from these components.

2.3. Feature extraction

2.3.1. Texture features

Texture feature is one of the most important attributes used in image analysis and pattern recognition. Gradient images are created from the original image by convolving with a filter, such as the popular Sobel filter [4], reflecting the directional change in the intensity or color in the image. To keep it simple, we assume that there are 360 directions, i.e., 0°, 1°, …, 360°. By recording the proportion of pixels in each direction, a histogram of gradient direction with 360 bins is created for each image. In order to reduce the number of bins in the histogram, every consecutive k (e.g., k=30) directions are combined to form one bin. Therefore, the histogram is represented by a vector gd with $360/k$ items.

A single level 2-dimensional wavelet decomposition is performed on the luminance layer in the CIE LUV space of the original image by using two channel filter banks composed of a low-pass and a high-pass filter in both horizontal and vertical directions. The original image is then decomposed into four wavelet coefficient images, i.e. the low-low (LL), low-high (LH), high-low (HL) and high-high (HH) channels [5, 6]. A gradient vector gd with $360/k$ features can be generated for each sub-image. Let gd_{LL}, gd_{LH}, gd_{HL}, and gd_{HH} represent the gradient vector of each channel, the texture feature vector of each spam image is the concatenation of gd_{LL}, gd_{LH}, gd_{HL}, and gd_{HH}.

2.3.2. Layout features

In our observation, spammers often produce images using the same foreground layout with slightly shifted foreground objects, or the same layout but with slightly different foreground graphics. Based on this observation, we may assume that spam images from the same origin share similar foreground spatial layout. However, the slightly shifted foreground objects may introduce severe noise in a foreground layout comparison. To alleviate such a problem, we extract a minimum bounding rectangle (MBR) which contains all the foreground objects in the foreground illustration mask produced in the segmentation step. The extracted MBRs are resized into the same dimensions (e.g., 150×150 pixels), and then divided into 9 subareas as a 3×3 grid as shown in Figure 4.

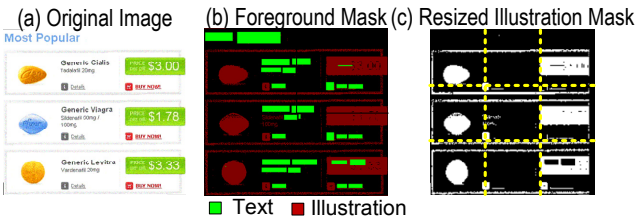

Fig. 4. Gridding the foreground mask of a spam image.

The layout feature is represented as the proportion of the foreground object pixels in each grid cell with the order from left to right and top to bottom.

2.3.3. Color features

In this study, we also adopt the 6-bit color-code histogram of foreground illustrations as color features, which can be extracted with the approach in [1].

2.3.4. Text features

Text features have proved to be robust and can tolerate the change of color, illustration, layout, and/or texture. Therefore, in this paper, we use OCR to extract the text content from spam images (see Figure 4(b)), and then, adopt Needleman-Wunsch algorithm as described in [1] to compute the similarity of text contents.

2.4. Phase I: clustering using visual features

Assume a D-dimensional dataset contains N data points. In our case, one data point represents the visual feature vector of a spam image, consisting of gradient direction features, foreground layout features, and color-code features. Each vector is represented as $x_i=(x_{i1}, x_{i2},\ldots, x_{iD})$, in which $1\le i\le N$, and x_{ij} ($1\le j\le D$) is the value of the j-th feature component of x_i. Since some components in an image's feature vector could be zero or empty, the $D\times N$ matrix (dataset) actually is a sparse matrix. A distance measure, such as Euclidean distance, calculates the similarity between data points by using all dimensions. Because of the *curse of dimensionality* and the sparsity of the matrix, irrelevant dimensions may be involved into the description of a cluster that exists in a subspace, making the distance measure ineffective.

In Phase I, a histogram-based projective clustering algorithm *REVBH* (Relative Entropy on Variable Bin-width Histogram), an improved version of the method in [7], is proposed to group spam images with foreground illustrations. *REVBH* consists of five steps: 1) constructing a variable bin-width histogram for each d-dimensional subspace; 2) detecting dense areas in each d-dimensional histogram; 3) converting each data point, i.e., a spam image, into a signature that describes how that data point is projected into different subspaces; 4) merging similar object signatures to form descriptions of clusters; 5) assigning data objects to clusters.

2.4.1. Construction of variable bin-width histogram

Since one variable bin-width histogram is created for each d-dimensional subspace (d is 2 in our study), there are ${}_DC_d$ histograms that need to be constructed for a given dataset. Each dimension is first partitioned into multiple sub-ranges according to the underlying data distribution. Then, different bin-widths are calculated for each sub-range. We further improve the partition strategy as follows:

First, a fixed bin-width histogram $hist_{org}$ (e.g., bin number=100) along each dimension is constructed as an approximation of the data distribution along that dimension (Figure 5(a)). Using polynomial curve fitting we find the local minimum points as the candidates for sub-range partitioning (Figure 5(b)). Sometimes no local minimum can

be found within the data range of a dimension for a specified degree n because of inadequate data samples. In such cases, the degree n which starts from 10 decreases by 1 iteratively until local minimum can be found.

Since in many cases, most data points may fall into a relatively narrow sub-range for some dimensions as shown in Figure 5(a), an equalized histogram $hist_{eq}$ is constructed based on the original histogram $hist_{org}$ by using a typical histogram equalization method [8] as shown in Figure 5(c). Then, the local minimum points within that narrow range are generated by using polynomial curve fitting (Figure 5(d)). In the end, two sets of local minimum points (the two sets from Figures 5(b) and (d)) are merged. If two local minimum points are too close with each other, one of them is removed. The rest of the local minimum points are treated as the partitioning boundaries of sub-ranges in the current dimension. In Figure 5(a), the vertical lines indicate the final partitioning boundaries along one dimension, in which red lines result from the original histogram, while the green margin results from the equalized histogram in Figure 5(d).

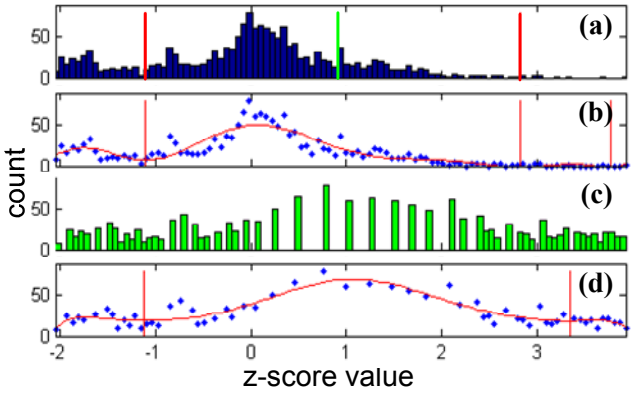

Fig. 5. Partition on one dimension by using original histogram and equalized histogram. (a) Original histogram $hist_{org}$ for one dimension; (b) Partition on the $hist_{org}$ with polynomial of degree 10; (c) Equalized histogram $hist_{eq}$; (d) Partition on the $hist_{eq}$ with polynomial of degree 10.

The bin-width of each sub-range along one dimension is determined by using Freedman and Diaconis's rule [9]:

$$h = 2 \times IQR \times n^{-1/3} \quad (1)$$

or Scott's rule [10]:

$$h = 3.5 \times \sigma \times n^{-1/3} \quad (2)$$

where IQR is the sample interquartile range, σ is the sample standard deviation, and n is the number of observations in the sample. Both rules are well-founded in statistical theory. Our strategy selects the fewer bins whichever method generates in each sub-range. The number of bins in each sub-range is calculated as:

$$binNum = length\ of\ sub\text{-}range / h \quad (3)$$

where h represents the selected bin-width in the current sub-range calculated by Equations (1) or (2).

In a d-dimensional ($0<d\leq D$) histogram, let φ denote its corresponding subspace and each dimension is φ_i ($i \in [1,d]$). The height of each bin of that histogram is calculated by:

$$height(bin_m) = \frac{number\ of\ objects\ in\ bin_m}{N \times (h_{\varphi_1, sub_{\varphi_1,m}} \times h_{\varphi_2, sub_{\varphi_2,m}} \times \ldots \times h_{\varphi_d, sub_{\varphi_d,m}})} \quad (4)$$

where N is the total number of data objects. m is the index of a bin ($1 \leq m \leq$ total number of bins in the d-dimensional histogram). $h_{\varphi_i, sub_{\varphi_i,m}}$ ($i \in [1,d]$) represents the bin-width of the $sub_{\varphi i,m}$-th sub-range of the dimension φ_i. The product $(h_{\varphi_1, sub_{\varphi_1,m}} \times h_{\varphi_2, sub_{\varphi_2,m}} \times \ldots \times h_{\varphi_d, sub_{\varphi_d,m}})$ is therefore the volume of that d-dimensional bin.

2.4.2. Relative entropy as a density threshold

Relative entropy [11] is a non-symmetric measure of the difference between two probability distributions P and Q. It represents the similarity between real distribution P of data and compared distribution Q, such as a uniform distribution, of the same data in a d-dimensional subspace. The more similar is a real distribution to a uniform distribution, the further relative entropy approaches 0. The relative entropy $H_r(X)$ of a d-dimensional histogram is defined as:

$$H_r(X) = \sum_{i=1}^{T} [p(x_i) \log_2 (p(x_i)/q(x_i))], |X| = T \quad (5)$$

where X represents the complete set of bins in the current histogram, $|x_i|$ is the number of objects in bin i, T is the total number of bins in the current histogram, $p(x_i)$ is the normalization of the height of bin i under real distribution, and $q(x_i)$ is the normalization of height of the same bin under uniform distribution.

$$p(x_i) = \frac{|x_i| / (N \times h_i)}{\sum_{j=1}^{T} [|x_j| / (N \times h_j)]} = \frac{|x_i| / h_i}{\sum_{j=1}^{T} (|x_j| / h_j)} \quad (6)$$

where $h_i = h_i^1 \times h_i^2 \times \ldots \times h_i^d$ is the volume of the i-th bin in the current d-dimensional histogram. h_i^k, $1 \leq k \leq d$, is the bin-width on the k-th dimension.

$$q(x_i) = \frac{N(h_i / S) / (N \times h_i)}{\sum_{j=1}^{T} [N(h_j / S) / (N \times h_j)]} = \frac{1/S}{T/S} = \frac{1}{T} \quad (7)$$

where $S = \sum_{j=1}^{T} h_j$ is the total area that contains data objects in the current histogram. Then, $H_r(X)$ can be represented as

$$H_r(X) = \sum_{i=1}^{T} p(x_i) \log_2 (T \cdot p(x_i)), |X| = T \quad (8)$$

Let the relative entropy of a single bin in a d-dimensional histogram be defined as:

$$h_r(x) = p(x) \log_2 (T \cdot p(x)), x \in X \quad (9)$$

It is assumed that a dense area is surrounded by sparse area(s) with uniform distribution. $(1/T)H_r(X)$ can be used as the threshold in each histogram to distinguish between the single bins with lower relative entropy h_{r_low} and the single bins with higher relative entropy h_{r_high}. Thus, relative entropy is used in a greedy algorithm to detect dense bins in a histogram. Adjacent dense bins are combined to form a larger dense area. Then, relative entropy $H_r(x)$, $p(x_i)$ and $q(x_i)$ of each remaining bin are updated after the removal of dense areas from the current histogram. The iteration terminates when $H_r(x)$ is smaller than a small threshold ε or

the updated $H_r(x)$ is larger than the previous one. ε (e.g. ε = 0.0001) is a cutoff value that indicates sufficient similarity between the real distribution and uniform distribution in the current subspace.

2.4.3. The remaining steps in REVBH

In Steps 3 and 4, according to whether data objects fall into one dense area in each histogram or not, a signature (by the definition in [12]) that represents dense areas in subspaces where the data object is located is generated for each data object. Those signatures which represent the same or similar subspaces are merged to form one signature. The merged signatures are then sorted in descending order of their weight which is determined by the size of their corresponding clusters. The higher the weight, the more data objects are located in the corresponding subspace.

In the end, each data object is associated with one signature with the highest similarity. Data objects having similarity with each signature lower than a threshold δ are treated as outliers. δ indicates the proportion of matched subspaces between a data point and a cluster.

2.4.4. Removing zero values from data points

Histogram-based clustering detects dense areas represented by bins with relatively high height in a histogram that corresponds to a subspace. Because of the sparsity of the data matrix as aforementioned, many zero/empty values may be observed on one feature dimension. It is possible that on some feature dimensions, dominant bins with zeros values exist and could be mis-detected as dense areas. However, those dominant zero-valued bins have almost no contribution to distinguishing between clusters but to introduce noise and bias. Therefore, in our framework, such bins in *d*-dimensional histograms are not considered for constructing subspaces.

2.5. Phase II: clustering using text features

In Phase II, a text-based clustering is used to group spam images based on the edit distance associated with Needleman-Wunsch algorithm. Text clue is used to further merge the clusters from Phase I with those images that contain mainly text but not illustrations. Specifically, this scheme merges two clusters if their text similarity is greater than 97% based on the single linkage method.

3. EXPERIMENTAL RESULTS

Spam images are identified manually from collected emails through the use of "catch all" email address. A "catch all" configuration accepts email for all possible addresses at a given domain. 2,100 spam images including 37 wavy images are processed and tested in our experiment. In order to evaluate clustering results, the ground truth consisting of 476 classes is manually collected. All features are normalized into z-score [13]. V-measure [14] and the number of produced clusters are both used to evaluate the clustering quality.

3.1. Parameter tuning

There are two main parameters in the first phase, namely the expected maximum number of clusters *max_no_cluster* and the similarity threshold δ between data points and clusters. To test the robustness of the our method under different parameter settings, the proposed 2-phase algorithm is tested with *max_no_cluster* $\in$ {500, 600, 700} and $\delta \in$ {0.8, 0.9, 1.0} as shown in Tables 1 and 2.

Table 1. Results with different δ values (*max_no_cluster*=500)

δ	V_1-*measure*	*Cluster #* (*class #*: 476)
0.8	0.9269	471
0.9	0.9315	471
1.0	0.9315	471

If *max_no_cluster* is larger than the number of natural, clusters, it will return all discovered clusters. Otherwise, only the top ranked *max_no_cluster* clusters are returned. δ=1 indicates that the data point falls exactly into the subspace of the cluster. We find that the V-measure and the number of clusters become stable when δ increases to 0.9 and above. Thus, we perform the remaining experiments with this parameter setting.

Table 2. Results with different *max_no_cluster* (δ=1.0)

Max #	V_1-*measure*	*Cluster #* (*class #*: 476)
500	0.9315	471
600	0.9316	472
700	0.9318	472

According to our experiment, when the expected maximum number of clusters is set to a large value, such as 600 or 700, more small clusters are generated. Those clusters are represented as signatures with low weights and seated near the end of the signature list. Some data points that were previously treated as outliers are now assigned to those small clusters. Table 2 shows that the text clustering in Phase II can recombine these small clusters into larger ones, and finally achieves a relatively stable number of clusters.

3.2. Effectiveness of wavy image correction

37 wavy images that are representative samples of >10,000 wavy images in our database with varying background and wavy distortions are all distorted horizontally. The effectiveness of the proposed wavy image correction algorithm is shown in Table 3. With wavy image correction, OCR can successfully extract the text content correctly from wavy images, thereby, improve the clustering quality.

3.3. Comparison with other algorithms

We compare the proposed framework with the hierarchical clustering algorithm in [15]. We test both algorithms on various feature combinations with *max_no_cluster*=500. Figure 6 shows that the highest V-measure (0.9315) with reasonable number of clusters (471) is achieved when performing the histogram-based projective clustering on all three visual features in Phase I, followed by text-based clustering in Phase II. Most V-measure values produced by hierarchical clustering are below 0.9. Since projective clustering considers only significant subspaces but not the full feature dimensions, it is less affected by the problem of features with dominant zero values, and thus, results in

better clustering results compared with the hierarchical clustering algorithm.

4. CONCLUSIONS

In this study, we present a novel, multi-clue spam image clustering framework which adopts the histogram-based projective clustering to group spam images based on foreground visual features and text content. The proposed approach eliminates the need to choose among distance metrics for clustering analysis, and thus, is more robust and efficient. In addition, a robust wavy image correction algorithm is proposed. OCR can effectively extract text from the corrected wavy images. Our experimental results demonstrate the effectiveness of the proposed framework.

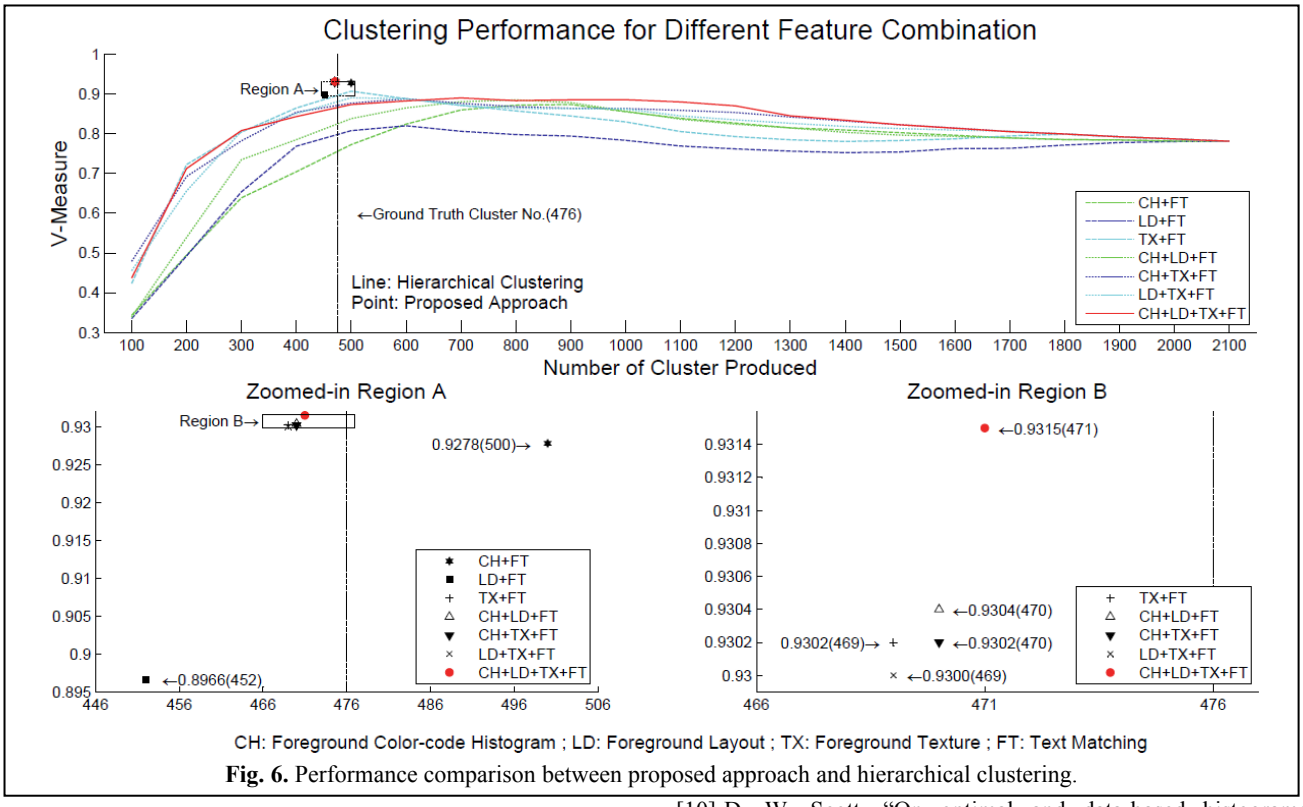

Fig. 6. Performance comparison between proposed approach and hierarchical clustering.

Table 3. Effectiveness of wavy image correction

Dataset	*V_1-measure*	*Cluster # (class #: 476)*
With corrected wavy image	0.9315	471
With original wavy image	0.9248	498

5. REFERENCES

[1] C. Zhang, W.-B. Chen, et al., "A multimodal data mining framework for revealing common sources of spam images", *J. of Multimedia*, vol. 4, no. 5, pp. 313-320, Oct. 2009.

[2] R. Rivest, "The md5 message-digest algorithm," RFC1321, 1992.

[3] L. S. Davis, "A survey of edge detection techniques," *Computer Graphics and Image Processing*, vol. 4, issue 3, pp. 248-270, Sep. 1975.

[4] R. M. Haralick and L. G. Shapiro, Computer and Robot Vision, vol. I, Addison-Wesley, Reading, MA, 1992.

[5] P. W. Huang, S. K. Dai, and P. L. Lin, "Texture image retrieval and image segmentation using composite sub-band gradient vectors," *J. of Visual Communication and Image Representation*, vol. 17, no. 5, pp. 947-957, Oct. 2006.

[6] M. F. A. Fauzi and P. H. Lewis, "Texture-based Image Retrieval Using Multiscale Sub-image Matching," *IS&T/SPIE's 15th Annual Symposium Electronic Imaging: Image and Video Communications and Processing* (SPIE 2003), pp. 407-416, Jan. 2003.

[7] C. C. Aggarwal and P. S. Yu, "Finding generalized projected clusters in high dimensional spaces," ACM SIGMOD 2000, pp. 70-81, 2000.

[8] http://en.wikipedia.org/wiki/Histogram_equalization

[9] D. Freedman and P. Diaconis, "On the histogram as a density estimator: L_2 theory," *Probability Theory and Related Fields*, vol. 57, no. 4, pp. 453-476, 1981.

[10] D. W. Scott, "On optimal and data-based histograms," *Biometrika*, vol. 66, no. 3, pp. 605-610, 1979.

[11] S. Kullback and R. A. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79-86, 1951.

[12] E. K. K. Ng, A. W. C. Fu, and R. C. W. Wong, "Projective clustering by histograms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 3, pp. 369-383, 2005.

[13] R. J. Larsen and M. L. Marx, "An introduction to mathematical statistics and its applications," Prentice Hall, 3rd Ed., pp. 282, 2000.

[14] A. Rosenberg and J. Hirschberg, "V-Measure: a conditional entropy-based external cluster evaluation measure," *Proc. of the 2007 Conf. on Empirical Methods in Natural Language Processing*, pp. 410-420, 2007.

[15] C. Zhang, X. Chen, et al., "Spam image clustering for identifying common sources of unsolicited emails," *Intl. J. of Digital Crime and Forensics*, vol. 1, no. 3, pp. 1-20, 2009.