Dynamic Print Stream Classification and Optimal JPEG Compression

Cheng Lu, Jan P. Allebach; Purdue University; West Lafayette, IN, U.S.A
Mark Q. Shaw, Randy Guay, David C. Day; Hewlett Packard Company, Boise, ID, U.S.A

Abstract

Image compression is the prerequisite for many applications. In some applications, different types of images may favor different compressions. For PC printing purpose, the system needs to choose optimal compression algorithm and parameters in order to obtain the best balance between image quality and compressed file size. For example, pure text with simple background image is suitable for lossless compression like Run Length Encoding, because it preserves the image quality while having small compressed file size. However, complex natural image may favor lossy compression like JPEG since it reaches good compression ratio at the price of image quality. In this case, we need to find an optimal compression level so that it reaches the best balance between the image quality and the compression ratio. In this paper, we propose a system that finds an optimal compression algorithm given the input image. Also, if the input image is decided to be compressed by the lossy compression (JPEG), the system will find the optimal compression level.

INTRODUCTION

Image compression is becoming increasing crucial for various purpose in today's information age [1, 2, 3, 4, 5]. One of these applications is image compression for printing[6, 7]. People nowadays print a variety of images or documents from either mobile devices or PC. When printing images or documents, there are two major factors which impact user experience: image quality [8, 9] and printer process time. Ideally, we hope to reach image quality as high as possible while process time as quick as possible. However, the two factors here can not be improved simultaneously. Improving image quality is typically at the expense of longer processing time because we need to deal with larger file size which is caused by less lossy compression [10]. Heavily lossy compressed images can be processed very fast by printer due to small file size, however it generates poor image quality outputs. So we propose a new approach which could jointly optimize both of these factors under the constrains given by current printer system.

The current printing system supports three compression modes: *Delta Row Compression* (DRC) [11], *Run Lenth Encoding(RLE)* [12] and *JPEG* [13]. In these three modes, DRC and RLE are lossless compression while JPEG is a lossy compression algorithm. Processing natural images using DRC or RLE is very inefficient. First, Human Vision System (HVS) is not sensitive to high frequency information in the image [14, 15], so it can not perceive some information loss in the lossy compression process. Secondly, using RLE and DRC will yield to very large file size for natural images, because typical natural images do not have much spacial redundancy [16, 17, 18]. These two reasons drive us to use lossy compression algorithm such as JPEG to process natural images. On the other hand, simple structure image is more suitable for DRC or RLE lossless compressions. Simple structure image such as small logo or pure text have very high spatial redundancy which can be compressed heavily by RLE or DRC with no cost of image quality. Plus, JPEG compression will blur edges of text [19], which is undesirable in case of simple structure image.

Given these facts and constraints, it is clear that the proposed system need to first distinguish if the input belongs to natural image or simple structure image. This requires a classifier which extracts certain features from the input images, then makes a binary decision. If the decision is a natural image, we feed the image to the JPEG compressor. Otherwise, if it is decided to be a simple structured image, we send it to one of lossless, RLE or DRC, compressors. For natural images, we need to control the image quality by choosing optimal Q factor [20]in JPEG compressor which reaches a good balance between output image quality and compressed file size. A larger Q factor yields to better image quality at the expense of larger file size.

Image quality is not our concern if the input is classified as simple structure image because both RLE and DRC are lossless compressions, which means they are equivalent in terms of image quality. However, they may differ in terms of decompression time at printer firmware. So we need to make choice between RLE and DRC based on only one metric: decompressed time at printer. Unfortunately, decompression algorithms are only available in printer firmware but not PC, so we need to develop an classification algorithm in PC or mobile devices which can predict the decompression time consumed by printer, if the input image needs to be losslessly compressed.

Content-based image classification [21, 22, 23], compression and image quality have been intensively studied separately. In this paper, we combine these three factors together to optimize the printing performance. This system can also apply to other platforms that require balance between image quality and compression ratio.

We applied the Support Vector Machine [24] for classification purpose at the first stage which distinguishes lossy and lossless compressions. With more than 7000 training images, we managed to build a robust SVM classifier. JPEG compressed image quality is a long-studied topic [25, 26, 27]. Most of these studies focus on finding correlation between human perceived image quality and certain features [25]. In this paper, we propose a Dynamic Print Stream Compression (DPSC) engine to finding the optimal compression level (Q factor) in JPEG compression, which could reach a good balance between image quality and compression ratio. This would largely enhance the PC printing performance. With DPSC engine, we could significantly improve the efficiency of printing. Proper compression for different types of input images could reduce the decompression time in the printer firmware while preserving good print quality. It makes sure that we get good print quality at the lowest price of decompression load.

PRINTING SYSTEM DESCRIPTION

As described in introduction, we have a printing system as shown in Fig. 1. Our proposed DPSC engine and three compressors are implemented in PC or mobile devices. Computationally, PC or mobile devices are much more powerful compared with printer firmware. So we can safely ignore processing time at PC end, which is shown to the left of dash line in Fig. 1. Then we assume that overall processing time is fully determined by decompression time consumed by printer firmware.

DPSC ENGINE

structure

Our proposed DPSC engine follows a hierarchical decision structure as shown in Fig. 2. Since we will only only use luminance channel information in the DPSC engine, we first transform the original image into LUV color space and keep only luminance channel. Using only luminance channel information will significantly speed up subsequent image classification and image quality analysis. At the first classification stage, DPSC engine decides if the input image should be compressed by lossy or lossless compressors. If the input is a natural image, we will use JPEG to compress it and find its optimal Q factor. If it is a simple structure image such as pure text or small logo, we apply the second stage classification between RLE and DRC depending on the prediction of the decompression time consumed by printer firmware.

lossy vs. lossless classification

As described in introduction , we first need to decide if the input image is a natural image or simple structure image, then apply compressor correspondingly. With more than 7000 training images, we develop a SVM classifier for the task. The input of this SVM classifier is a 3-dimensional feature vector. The three elements in the feature vectors are *Histogram Flatness, Histogram Span and Luminance Variablity Score*. We use these three features to train our first lossy vs. lossless SVM classifier.

histogram flatness

If we build a histogram for a simple structure image such as pure text or logo, we can expect that this histogram is very peaky. There are roughly two peaks in this kind of histogram, one for text pixels and the other one for background pixels. However, if build the same histogram for a natural image, it should be more flat and widespread. A typical example of the histogram difference is given in Fig. 3. In order quantify this difference, we define the histogram flatness as its geometric average over arithmetic average

$$Flatness = \frac{\sqrt[N]{\prod_{n=0}^{N-1} x(n)}}{\frac{\sum_{n=0}^{N-1} x(n)}{N}} \tag{1}$$

where $x(n)$ is the number in bin n.

histogram span

The first feature may not work very well when histogram is relatively sparse. For example, if we have a histogram which satisfies $X(2n) = k$ and $X(2n+1) = 0$, its corresponding image should be closer to natural but rather simple structure. However, the first feature would decide the image is very peaky. To solve this issue, we develop the second feature $HistogramSpan$. It is defined as width of the smallest interval that includes %75 pixels.

luminance variability score

This feature is developed in [17]. This feature is based on the fact that the nontext region of a text image typically contains only a few gray level value. We cut the input image into 8×8 pixel blocks and calculate the mean value of the block. Then we build a 16-bin histogram for these block-mean values in the entire image. The Luminace Variability Score is defined as the number of non-zero bins in this block-mean histogram.

LOSSLESS CLASSIFICATION

Run Length Encoding and Delta Row Compression are two widely used lossless compression algorithm. As described in Sec. , we only need to consider the decompression time in the printer firmware.

Here we describe how we generate ground truth to train this classifier. For all the images which are labeled as simple structure in the training set, we compressed and decompressed it by both DRC and RLE. Then we measure its decompression time T^d_{DRC} and T^d_{RLE} respectively. If $T^d_{DRC} < T^d_{RLE}$, we label this image as DRC, otherwise we label it as RLE. This gives a two-class training set to train the classifier.

The most intuitive solution to finding faster decompression in printer firmware is doing both RLE and DRC decompression at PC or mobile end, then choose the faster one. However, due to our system constraint, we are not allowed to implement decompression algorithm on PC or mobile devices. So we need to resort to other features to do this prediction on PC or mobile end.

Although we can not apply decompression on PC or mobile, we can still compress the input image in both ways and find good predictors in the compression process. Two feature we find useful are *compression time ratio (CTR)* and *compression size ratio (CSR)*

For every input image, we compress it by both RLE and DRC. We measure compression time T^c_{RLE} and T^c_{DRC} respectively. Also, we measure the compressed file size F_{RLE} and F_{DRC}. Two features are defined as

$$CTR = \frac{T^c_{RLE}}{T^c_{DRC}} \tag{2}$$

$$CSR = \frac{F_{RLE}}{F_{DRC}} \tag{3}$$

With training set and features above, we can train a SVM classifier which is able to classify simple structure image into RLE or DRC.

OPTIMAL JPEG COMPRESSION

If the input image is classified as a natural image, we will use JPEG to compress it. In this case, we want to compress it

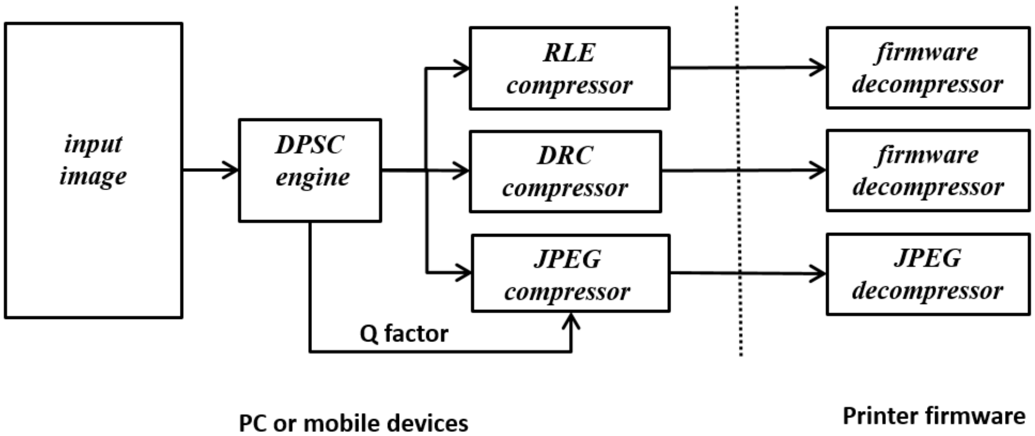


Figure 1: Printing system structure

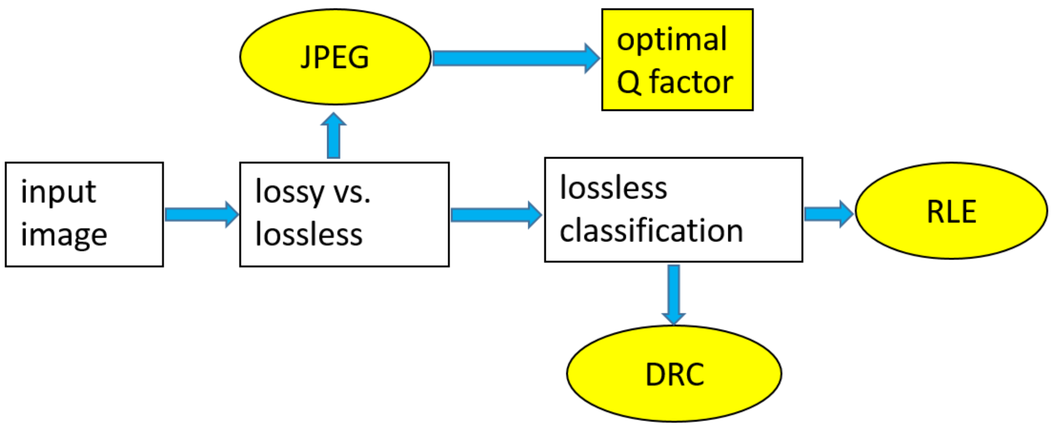
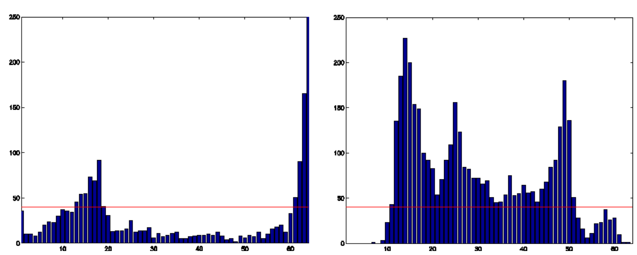


Figure 2: DPSC engine structure



(b) Simple structure image histogram (c) Natural image histogram

Figure 3: Natural and simple structure image histogram comparison

as heavily as possible to reach smaller file size. However, if we compress it too hard, we will have very poor image quality output. In order to reach a balance between the compressed file size and output image quality, we need to find an optimal Q factor which is a variable in JPEG that controls the compression ratio and image quality.

However, Q factor does not change linearly with either compression ratio or image quality. [25] introduces three features that can quantify the JPEG image quality. All of these features are calculated horizontally and then vertically. The three features are *average differences across block boundaries* (B), *in-block absolute difference* (A) and *zero-crossing rate* (C).

If we have a image signal as $x(m,n)$ for $m \in [1,M]$ and $n \in [1,N]$, and calculate a difference signal along each horizontal lines:

$$d_h(m,n) = x(m,n+1) - x(m,n), \qquad x \in [1, N-1] \tag{4}$$

Average differences across block boundaries shows blockiness effect caused by JPEG compression, and it is defined as

$$B_h = \frac{1}{M(\lfloor N/8 \rfloor - 1)} \sum_{i=1}^{M} \sum_{j=1}^{\lfloor N/8 \rfloor - 1} |d_h(i, 8j)| \tag{5}$$

The other two features are related to the activity of the image signal. The activity is measured by two factors. The first is in-block absolute difference which is defined as

$$A_h = \frac{1}{7}\left[\frac{8}{M(N-1)} \sum_{i=1}^{M} \sum_{j=1}^{N-1} |d_h(i,j) - B_h|\right] \tag{6}$$

The second activity measure is the zero-crossing rate. We first define

$$z_h(m,n) = \begin{cases} 1, & \text{horizontal zero-crossing at } d_h(m,n) \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

The horizontal zero-crossing rate then can be estimated as:

$$Z_h = \frac{1}{M(N-2)} \sum_{i=1}^{M} \sum_{j=1}^{N-2} Z_h(m,n) \tag{8}$$

Similarly, we can get vertical features B_v, A_v and Z_v. We average over horizontal and vertical features to get the overall features;

$$B = \frac{B_h + B_v}{2},\ A = \frac{A_h + A_v}{2},\ Z = \frac{Z_h + Z_v}{2}. \tag{9}$$

The final prediction of Mean Opinion Score (MOS) can be calculated using the above three features as

$$MOS = \alpha + \beta B^{\gamma_1} A^{\gamma_2} C^{\gamma_3}, \tag{10}$$

Figure.4 shows the result of MOS prediction trained by two groups of images.

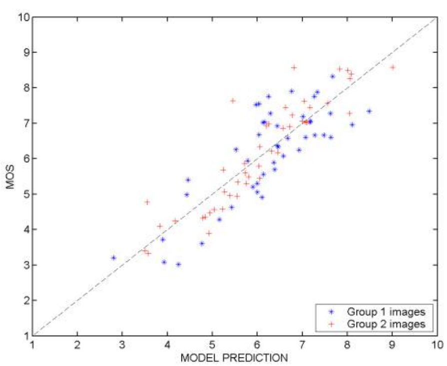


Figure 4: MOS prediction trained by two groups of images

We set a threshold Mean Opinion Score MOS_T depending on how good image quality we want to reach.

For any input natural image, we compress it starting from $Q_i = 10, 20, 30...., 100$. At each Q_i, we extract 3 features of compressed image as described in [25]. Then we calculate the predicted MOS_i at Q_i based on these 3 features. We find the first Q_i which has $MOS_i > MOS_T$, and set $i^* = i$. Thus, the optimal Q factor is defined as

$$Q^* = Q_{i^*} \tag{11}$$

Then we send the JPEG compressed image at the Q^* to printer.

EXPERIMENTAL RESULTS

lossy vs. lossless classification

To train the lossy vs. lossless classifier, we generate 8565 images from print drive engine. Among these 8565 images,4535 of them are labeled as natural images which ideally should be sent to lossy compressor (JPEG), while the other 4030 images are simple structure images which should be sent to lossless compressors (RLE or DRC). We use standard F_1 metric [**?**] to evaluate the performance of this binary classification problem. 4-fold cross validation is conducted in our experiment. The SVM classifier utilize the RBF kernl. Subsequently, the best F_1 score, precision, recall are shown in Table. 1, and its corresponding confusion matrix is given in Table. 2. In order to visualize this classification, we show the feature distribution of the image set in Fig. 5.

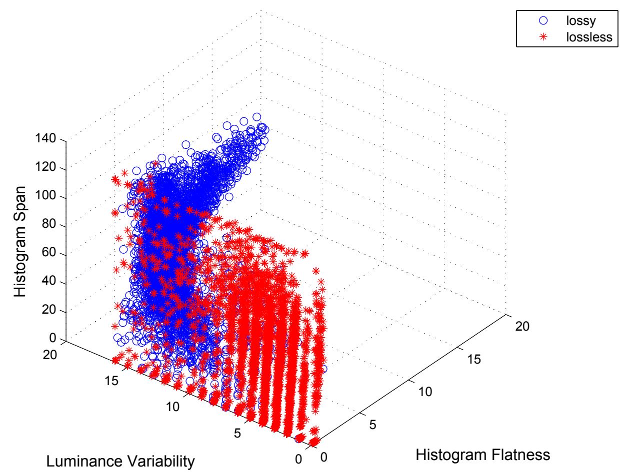


Figure 5: lossy and lossless classification in feature space

Table 1: Best F_1 score in cross validation for lossy vs. lossless

metric	precision	recall	F_1^*
data	0.946	0.898	0.924

Table 2: Confusion matrix at F_1^* for lossy vs. lossless

outcome / groundtruth	lossy	lossless
lossy	4291	244
lossless	488	3532

lossless classification

To train the RLE vs. DRC classifier, we have a training set with 4027 simple structure images. They are either pure text documents or simple logo image patches. As described in Sec. , we labeled these simple structure images based on their decompression time. Each image is labeled as the algorithm which is faster to decompress. Similar to lossy vs. lossless classification evaluation , 4-fold cross validation is conducted. And we also use RBF kernel in the SVM classifier. The best F_1 score, precision, recall are shown in Table. 3, and its corresponding confusion matrix is given in Table. 4. In order to visualize this classification, we show the feature distribution and decision boundary of the training set in Fig. 6.

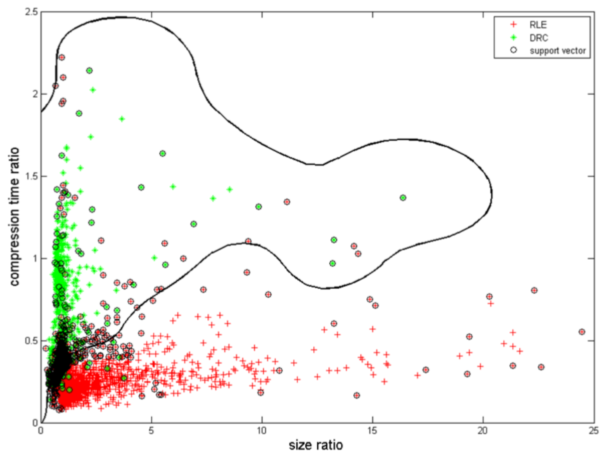


Figure 6: RLE and DRC classification in feature space and its decision boundary

Table 3: Best F_1 score in cross validation for RLE vs. DRC

metric	precision	recall	F_1^*
data	0.899	0.967	0.932

Table 4: Confusion matrix at F_1^* for RLE vs. DRC

outcome / groundtruth	lossy	lossless
lossy	3075	344
lossless	104	504

CONCLUSION

In this paper, we propose a dynamic printing stream compression engine which is able to choose best strategy to compress the input image. The DPSC engine follows a hierarchical decision structure. It first decides if the input is a natural image or simple structure image. The natural image will be sent to JPEG compressor while the simple structure image will be compressed losslessly. The second classifier will decide if the simple structure image should be compressed by RLE or DRC based on prediction of decompression time in each way. If the image is decided to be compressed by JPEG, DPSC engine will choose the optimal Q factor to reach good balance between image quality and compressed file size.

References

[1] David Taubman and Michael Marcellin, *JPEG2000 Image Compression Fundamentals, Standards and Practice: Image Compression Fundamentals, Standards and Practice*, vol. 642, Springer Science & Business Media, 2012.

[2] Marcelo J Weinberger, Gadiel Seroussi, and Guillermo Sapiro, “The loco-i lossless image compression algorithm: principles and standardization into jpeg-ls,” *Image Processing, IEEE Transactions on*, vol. 9, no. 8, pp. 1309–1324, 2000.

[3] Athanassios Skodras, Charilaos Christopoulos, and Touradj Ebrahimi, “The jpeg 2000 still image compression standard,” *Signal Processing Magazine, IEEE*, vol. 18, no. 5, pp. 36–58, 2001.

[4] Adrian S Lewis and G Knowles, “Image compression using the 2-d wavelet transform,” *Image Processing, IEEE Transactions on*, vol. 1, no. 2, pp. 244–250, 1992.

[5] Edward J Delp and O Robert Mitchell, “Image compression using block truncation coding,” *Communications, IEEE Transactions on*, vol. 27, no. 9, pp. 1335–1342, 1979.

[6] Amir Said and William A Pearlman, “Reversible image compression via multiresolution representation and predictive coding,” in *Visual Communications’ 93*. International Society for Optics and Photonics, 1993, pp. 664–674.

[7] Patrick Haffner, Paul G Howard, Patrice Simard, Yoshua Bengio, Yann Lecun, et al., “High quality document image compression with djvu,” *Journal of Electronic Imaging*, vol. 7, no. 3, pp. 410–425, 1998.

[8] Zhou Wang and Alan C Bovik, “A universal image quality index,” *Signal Processing Letters, IEEE*, vol. 9, no. 3, pp. 81–84, 2002.

[9] Zhou Wang, Alan Conrad Bovik, Hamid Rahim Sheikh, and Eero P Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *Image Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 600–612, 2004.

[10] Vivek K Goyal, Alyson K Fletcher, and Sundeep Rangan, “Compressive sampling and lossy compression,” *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 48–56, 2008.

[11] Gary L Vondran Jr, “Method and apparatus for delta row decompression,” Sept. 19 1995, US Patent 5,452,405.

[12] Stuart C Hinds, James L Fisher, and Donald P D’Amato, “A document skew detection method using run-length encoding and the hough transform,” in *Pattern Recognition, 1990. Proceedings., 10th International Conference on*. IEEE, 1990, vol. 1, pp. 464–468.

[13] Gregory K Wallace, “The jpeg still picture compression standard,” *Communications of the ACM*, vol. 34, no. 4, pp. 30–44, 1991.

[14] Jeffrey Lubin, “A human vision system model for objective picture quality measurements,” in *Broadcasting Convention, 1997. International*. IET, 1997, pp. 498–503.

[15] Gordon E Legge and John M Foley, “Contrast masking in human vision,” *JOSA*, vol. 70, no. 12, pp. 1458–1471, 1980.

[16] Marco Tagliasacchi, Alan Trapanese, Stefano Tubaro, João Ascenso, Catarina Brites, and Fernando Pereira, “Exploiting spatial redundancy in pixel domain wyner-ziv video coding,” in *Image Processing, 2006 IEEE International Conference on*. IEEE, 2006, pp. 253–256.

[17] Xiaogang Dong, Kai-Lung Hua, Peter Majewicz, Gordon McNutt, Charles A Bouman, Jan P Allebach, and Ilya Pollak, “Document page classification algorithms in low-end copy pipeline,” *Journal of Electronic Imaging*, vol. 17, no. 4, pp. 043011–043011, 2008.

[18] Frank HP Fitzek and Martin Reisslein, “Mpeg-4 and h. 263 video traces for network performance evaluation,” *Network, IEEE*, vol. 15, no. 6, pp. 40–54, 2001.

[19] Pina Marziliano, Frederic Dufaux, Stefan Winkler, and Touradj Ebrahimi, “A no-reference perceptual blur metric,” in *Image Processing. 2002. Proceedings. 2002 International Conference on*. IEEE, 2002, vol. 3, pp. III–57.

[20] Ci Chen, Yun Q Shi, Weijie Chen, and Guorong Xuan, “Statistical moments based universal steganalysis using jpeg 2-d array and 2-d characteristic function,” in *Image Processing, 2006 IEEE International Conference on*. IEEE, 2006, pp.

105–108.

[21] Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein, "Textural features for image classification," *Systems, Man and Cybernetics, IEEE Transactions on*, , no. 6, pp. 610–621, 1973.

[22] Jianchao Yang, Kai Yu, Yihong Gong, and Tingwen Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 1794–1801.

[23] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong, "Locality-constrained linear coding for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 3360–3367.

[24] Corinna Cortes and Vladimir Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[25] Zhou Wang, Hamid R Sheikh, and Alan C Bovik, "No-reference perceptual quality assessment of jpeg compressed images," in *Image Processing. 2002. Proceedings. 2002 International Conference on*. IEEE, 2002, vol. 1, pp. I–477.

[26] R Venkatesh Babu, Sundaram Suresh, and Andrew Perkis, "No-reference jpeg-image quality assessment using gap-rbf," *Signal Processing*, vol. 87, no. 6, pp. 1493–1503, 2007.

[27] Ahmet M Eskicioglu and Paul S Fisher, "Image quality measures and their performance," *Communications, IEEE Transactions on*, vol. 43, no. 12, pp. 2959–2965, 1995.