mPDF: Framework for Watermarking PDF Files using Image Watermarking Algorithms

Sachin Mehta*, Balakrishnan Prabhakaran, Rajarathnam Nallusamy, and Derrick Newton

Abstract

The advancement in digital technologies have made it possible to produce perfect copies of digital content. In this environment, malicious users reproduce the digital content and share it without compensation to the content owner. Content owners are concerned about the potential loss of revenue and reputation from piracy, especially when the content is available over the Internet. Digital watermarking has emerged as a deterrent measure towards such malicious activities. Several methods have been proposed for copyright protection and fingerprinting of digital images. However, these methods are not applicable to text documents as these documents lack rich texture information which is abundantly available in digital images. In this paper, a framework (mPDF) is proposed which facilitates the usage of digital image watermarking algorithms on text documents. The proposed method divides a text document into texture and non-texture blocks using an energy-based approach. After classification, a watermark is embedded inside the texture blocks in a content adaptive manner. The proposed method is integrated with five known image watermarking methods and its performance is studied in terms of quality and robustness. Experiments are conducted on documents in 11 different languages. Experimental results clearly show that the proposed method facilitates the usage of image watermarking algorithms on text documents and is robust against attacks such as print & scan, print screen, and skew. Also, the proposed method overcomes the drawbacks of existing text watermarking methods such as manual inspection and language dependency.

Keywords

Watermarking, Text Documents, PDF, Data Hiding, Print Screen

I. Introduction

The advancement in digital technology has revolutionized traditional business models. Nowadays, consumers can create, share, and distribute digital content across the world in fractions of a second. On one hand, these advancements have made human life easy, but on the other hand, it has increased piracy rates. Pirated digital content is readily available over the Internet.

Digital Rights Management (DRM) based systems are available for protecting content from misuse. These systems restrict access to the content and protect the document from misuse until it is encrypted or access restricted [1]. Once the document is obtained in clear text form, these systems might not be able to protect digital rights [2]. A simple example for creating a nearly similar copy of content is using *print screen*. A malicious user can take snapshots of all the pages of the document and construct a nearly similar copy which can be distributed illegally. Some DRM based systems disable such options while the content is being viewed, but these systems are very expensive [3]. Also, these systems are difficult to implement in an organization having heterogeneous networks. Apart from their own network, these organizations have other networks, usually of their clients. In such a case, organizations fail to use the features of DRM solutions and fail to protect the content.

Motivation and Contributions: Digital watermarking has emerged as a deterrent measure to protect the digital rights. Researchers have proposed several methods for copyright protection and traitor tracing of digital images. However, the application of these methods on text documents poses different challenges.

Digital watermarking aims at changing the pixel values such that changes are unnoticeable under normal viewing conditions. Pixels in images (either grey scale or color) take values from a wide range. For most pixels, the distortions resulting due to watermarking remain unnoticeable. Unlike images, pixels in text documents take values from a few possibilities. Due to lack of rich texture information in text documents, hiding data using image watermarking algorithms becomes difficult [4]. In such cases, watermark may be either lost or lead to visible distortions, as shown in Figure 1. To avoid such scenarios, researchers have proposed different methods for text watermarking and these are discussed in Section II. These methods suffer from several drawbacks such as limited applications and manual introspection.

In this paper, we propose an approach which overcomes the drawbacks of existing methods. Figure 2 provides an overview of the proposed method. We propose a framework which facilitates the usage of known image watermarking methods on text documents. To the best of our knowledge, there has been no work on developing techniques to watermark the text documents

Sachin Mehta is a graduate student at Department of Electrical Engineering, University of Washington, Seattle, USA. (e-mail: sacmehta@uw.edu)

Balakrishnan Prabhakaran is a Professor with Department of Computer Science, University of Texas at Dallas, Richardson, TX75080 USA (email:bprabhakaran@utdallas.edu)

Rajarathnam Nallusamy is a Director at Embnology Solutions Private Limited, Bangalore - 560037 (e-mail: raja@embnology.com)

Derrick Newton is a Lecturer at Faculty of Engineering and Computing, Coventry University, UK (email:ab3729@coventry.ac.uk)

This work was done as a part of when Sachin Mehta and Rajarathnam Nallusamy were working with Infosys Research Lab, Bangalore, India - 560100.

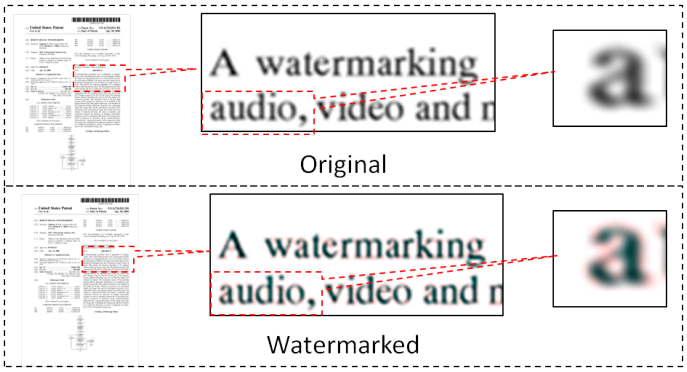


Fig. 1. Impact of image watermarking algorithms on text documents

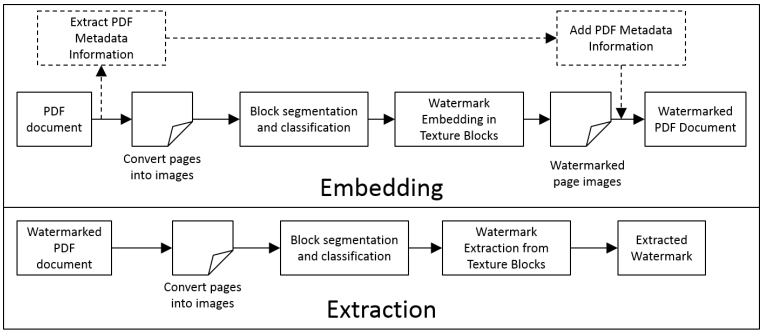


Fig. 2. Overview of the proposed method

using image watermarking algorithms. In particular, we propose an energy-based block classification approach which classifies the blocks into different categories (texture and non-texture blocks). A watermark is then embedded inside texture blocks using a content adaptive watermark embedding strength. We created a data set of text documents from 11 different languages and studied the imperceptibility, capacity, and robustness of 5 known image watermarking algorithms.

Rest of this paper is organized as: Related work is discussed in Section II. The proposed method is discussed in Section III. Criteria to evaluate the performance of the proposed method is given in Section IV. Experimental results are discussed in Section V. Conclusions are drawn in Section VI.

II. Related Work

Text watermarking methods can be classified into the following categories:

1) **Character Feature Methods:** These methods manipulate the features of characters such as shape, size, and position. Wu et al. [4] proposed a flippable pixels based data hiding method. The method reads each page of the document as an image and determines the flippable pixels using a set of rules. These flippable pixels are then manipulated for embedding the watermark. Wenyin and Ningde [5] proposed a watermarking method in which specific characteristics of Chinese characters are used for embedding the watermark. The method detects Chinese characters with occlusive components (characters with one or more hollow closing regions) and uses these characters for embedding the watermark. A color quantization based watermarking method was proposed in [6] [7]. Based on the watermark bit, the method changes the color of the character. Character feature methods, in general, exploit the characteristics of the alphabets in any language for embedding the watermark, for example occlusive components in Chinese alphabets. These methods are language specific and have applications limited to tamper detection and document identification. If a different watermark is embedded using these methods for fingerprinting, then a malicious user can easily detect or destroy the watermark by a simple comparative operation between two different copies of same content.
2) **Open Space Methods:** These methods embed the watermark by modulating either inter-line or inter-word or inter-character space. Huang et al. [8] and Alattar et al. [9] proposed a text watermarking method in which inter-word spaces and inter-line distances are modified for embedding the watermark.
These methods have high embedding capacity. Like character feature methods, these methods have limited applications and cannot be used in fingerprinting.

3) **Zero Watermarking Methods:** Instead of embedding a watermark, these methods construct a watermark using text document features such as occurrence of words with four or more characters. Zhang et al. [10] proposed a text watermarking algorithm in which features such as high frequency words and sentence length are used to create a watermark. Yawai and Hiransakolwong [11] proposed a zero watermarking method using line intersection. The method determines the intersection point for each character which is then used as a watermarking point.
These methods construct the watermark from the document itself and hence, cannot be used for fingerprinting applications.
4) **Natural Language Watermarking (NLW):** These methods replace the words by their synonyms or sentences are transformed via suppression or inclusion of noun phrases. Topkara et al. [12] proposed a NLW method. To embed the watermark, the method transforms the sentence. Halvani et al. [13] proposed a NLW method for German language. The method embeds the watermark by lexical and syntactic substitution.
NLW methods are syntax and semantic based. A manual inspection is required for validating the accuracy of these methods. Further, these methods are difficult to extend for fingerprinting applications.
5) **Visible Watermarking:** These methods embed the visible watermark. Microsoft's word also offers this feature. As the watermark is visible, the consumers of the document can be easily identified. However, these methods cannot be used for forensic applications as a malicious user may try to frame an innocent user.

Existing methods suffer from following drawbacks:

- *Limited Applications:* These methods can be used in authentication, tamper detection, and copyright protection. These methods are not suitable for fingerprinting which aims at embedding a unique watermark for every consumer.
- *Multi-lingual Support:* Most of these methods utilize the specific characteristics of a particular language for watermarking which make their application to other language documents very difficult.
- *Manual Introspection:* NLW methods are based on substitution either at sentence or word level. Sometimes substitution may lead to a change in the meaning of the sentence. Hence, every watermarked document needs to be manually inspected. This is a tedious process and make the method practically infeasible.

This paper proposes a method which addresses the drawbacks of the existing methods such as limited applications and language dependency.

III. The Proposed Method

For addressing the challenges involved in adopting image watermarking strategies for text documents, we propose two main techniques: (i) content-based block classification; (ii) determination of content adaptive watermarking strength.

A. Content based block classification

The proposed method segments the page into blocks and selects a few blocks using the selection protocol to embed the watermark.

1) Block Analysis: Text document contains margins. These margins do not contain any information. The proposed method determine these margins before segmenting the document into blocks. Once the information about margins is obtained, the page is cropped and then, segmented into blocks which are classified using the selection protocol.

- **Detection of Page Margins and Cropping:** Let us assume that page P of a text document has dimension $N \times M$. Margins in a document can be easily detected in a text document. A simple way to detect the margin is to identify a first *non-white* pixel from all the sides of page. The distance of the first non-white pixel from the document edge denotes the margin. However, this simple approach fails when some distortions are present in the document. Intentional or un-intentional modification of the text document may lead to change in certain pixel values, say in compression. These changes will lead to false identification of margins. Figure 3 shows an example of false margin detection. Difference in dimension of the cropped page with and without distortion clearly indicates that the simple approach fails to detect margins correctly when distortions are present.
  To overcome such problems, we pre-process the page before identification of margins. The proposed method applies a discrete differentiation operator on the page. The proposed method uses a SOBEL operator for differentiation. For page P with coordinates (x, y), the magnitude of the gradient can be expressed as:

$$mag(\bigtriangledown P) = \sqrt{|G_x^2 \ + \ G_y^2|}, \ where \ \ G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * P, \ \ G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 2 & 1 \end{bmatrix} * P \tag{1}$$

  Once output of differentiation operator is obtained, distance of first *white pixel* is computed from all directions[1]. These distances can be represented as d_l, d_r, d_t, and d_b for left, right, top, and bottom sides respectively. Once distances are

[1]The proposed method doesn't apply any thresholding to the output of the differentiation operator

determined, page P is cropped to obtain the cropped page P_C having dimension $N_C \times M_C$. An example of the differentiation operator and cropping is shown in Figure 4.

- **Segmentation and Classification of Blocks:** Once the cropped page P_C is obtained, it is segmented into blocks of dimension $n \times m$. Let us say that b such blocks exist such that $P_C = \{B(1), B(2), ..., B(b)\}$ where $B(\cdot)$ represents the block. These b blocks are classified into two categories: (i) texture blocks and (ii) non-texture blocks. Texture blocks are the ones which contain either complete text (CT) or complete graphics (CG) or partial text and partial graphics (PTPG). Non-texture blocks are either completely white (CW) or completely black (CB) or blocks with partial texts (PT). Examples of different types of blocks are shown in Figure 5. The proposed method classify the blocks using energy E of the block. Figure 6 shows the classification of blocks using energy in gray level and RGB color space [2].
  The proposed method uses DCT to compute the energy of a block. This choice is made due to the *energy compaction* property of DCT where in the energy of the entire block is concentrated in a DC component and a few AC components. Blocks in text documents do not have rich texture information and hence, the majority of the energy of the block lies in DC coefficient. The proposed method classifies the block using the DC coefficient and the classification criteria is explained in Algorithm 1. The performance of Algorithm 1 is dependent upon the thresholds. Figure 7 shows the relation between the probability of false classification, block size, and thresholds. From Figure 7, it is clear that probability of false classification of blocks is less when $\gamma_1 = 0.9$, $\gamma_2 = 0.7$,$\gamma_3 = 0.4$, and $\gamma_4 = 0.1$. We have used the same values in our experiments [3].

Algorithm 1: Energy-based classification of Blocks

$B_{DCT} = dct(B(i)), \quad 1 \leq i \leq b$

$DC = B_{DCT}(0,0)$

if $DC > T_1$ *or* $DC < T_4$ **then**
  Non-texture block
else
  if $DC \leq T_1$ *and* $DC > T_2$ **then**
    Complete text block
  else if $DC \leq T_2$ *and* $DC > T_3$ **then**
    Partial graphics and partial text
  else if $DC \leq T_3$ *and* $DC > T_4$ **then**
    Complete Graphics

where

T_1, T_2, T_3, and T_4 are content thresholds used for block classification.

$T_1 = \gamma_1 \times B_{max}$, $T_2 = \gamma_2 \times B_{max}$, $T_3 = \gamma_3 \times B_{max}$, and $T_4 = \gamma_4 \times B_{max}$

$B_{max} = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{m} 255^2}{n \times m}}$ = maximum possible energy of the block

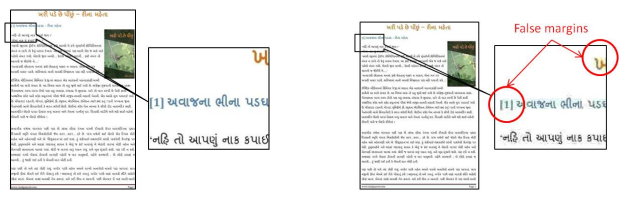
γ_1, γ_2, γ_3, and γ_4 are the constants used for setting the threshold values.

B. Watermarking

- **Content adaptive watermark embedding strength:** Once the blocks are classified, we need to embed the watermark inside these blocks. Fixed watermarking strengths may either lead to perceptibility (as shown in Figure 1) or automatic destruction of the watermark, as in completely white blocks where all the pixels in RGB color space have value (255, 255, 255). Watermark is like a noise and it's addition will increase the pixel value from (255, 255, 255) to (255 + Δ, 255 + Δ, 255 + Δ) assuming watermark is added uniformly across RGB channels. Here, Δ represents the change in pixel value due to watermark addition. Since a pixel can have a maximum value of (255, 255, 255) in RGB color space, the altered pixel value i.e. (255 + Δ, 255 + Δ, 255 + Δ) will be truncated back to the (255, 255, 255) leading to automatic destruction

[2]For illustration purpose, we have assumed that all pixels in a block contain same values.

[3]Histogram-based approach can classify the blocks only into two categories i.e. text and graphic blocks and is discussed in Appendix VII-A.



(a) 1361 × 1952 (b) 1403 × 1975

Fig. 3. False margin detection: (a) Cropped page without compression (b) Cropped page with compression (Quality factor=10)

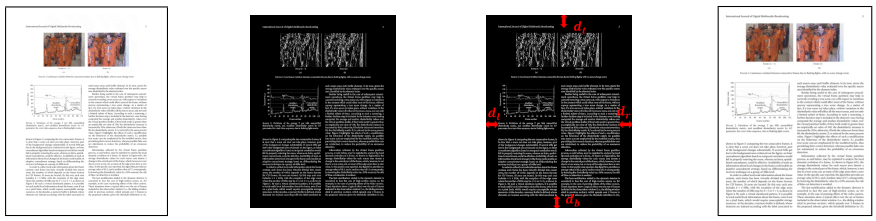


Fig. 4. Differentiation operation and cropping: (a) Original Page P, (b) Output of SOBEL operator, (c) Distances from different directions, and (d) Cropped page P_C

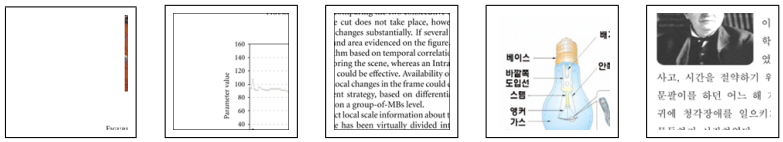


Fig. 5. Different types of blocks: (a -b) Non-texture blocks and (c-e) Texture blocks

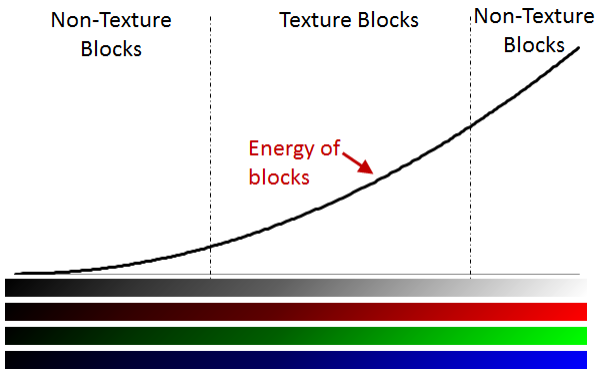


Fig. 6. Energy based classification of blocks [x-axis denotes the pixel values of the block while y-axis denotes the energy of the block.]

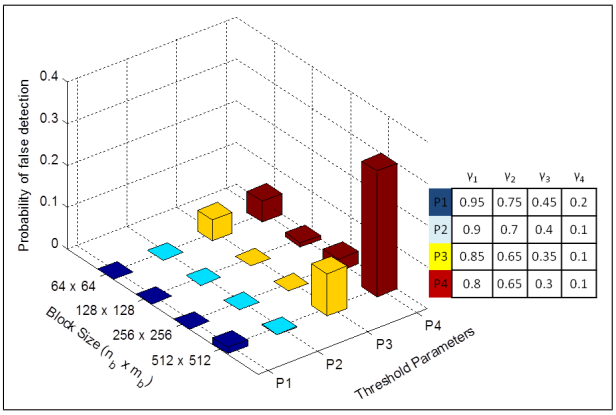


Fig. 7. Relationship between block classification, block size, and thresholds

of watermark. Considering these challenges, the proposed method embeds the watermark inside the "texture blocks" only using a content adaptive watermarking strength.

The energy of each block is different. The watermark embedding strength will vary for each block. Generally, the watermark embedding strength α ranges from 0 to 1 and is dependent upon *imperceptibility* and *robustness*. Texture blocks with high as well as low energy will have little embedding capacity and hence, the watermark embedding strength should be less while texture blocks with moderate energy have comparatively high embedding strength and hence, the watermark embedding strength should be relatively high. In summary, watermark strength should be adaptive to the content. A few watermark embedding curve examples (continuous as well as staircase functions) are shown in Figure 8. In the proposed method, we have used a staircase function. We embedded the watermark using DWT-based image watermarking algorithm at different α values for determining the watermark embedding strength for each category of block. After embedding a watermark, we manually inspected the watermarked blocks for imperceptibility while extracted the watermark using DWT-based image watermarking algorithm for checking the robustness. Results are summarized in Table I. From Table I, we can see that $\alpha = 0.1$ for CT blocks and $\alpha = 0.2$ for PTPG as well as CG blocks lead to watermark detection while being imperceptible. Hence, we have used these values for embedding the watermark inside texture blocks. As non-texture blocks lead to watermark loss, we have set $\alpha = 0$ for these blocks i.e. for CW and PT blocks. Further, we have tested these values of α with other image-based watermarking algorithms (see Table II) and found that all the image watermarking algorithms are able to detect the watermark with above specified values of α.

- **Watermark Extraction:** To extract the watermark from suspected document files, the proposed method converts the document into pages. Let us say that P' pages having dimension $N' \times M'$ exist in the suspected document file. The proposed method detects margins in P' pages and generates cropped pages P'_C using the method discussed in Section

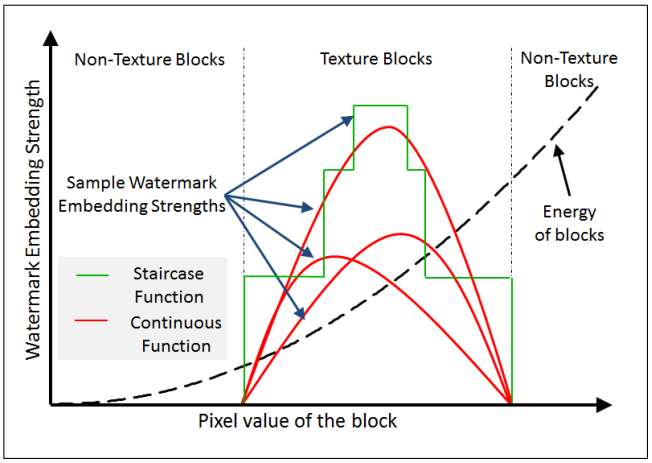


Fig. 8. Possible functions of watermark strengths (for the sake of illustration, we assume that block is homogenous i.e values of all pixels inside the block are same.)

TABLE I. DETERMINATION OF α

α	Non-texture blocks		Texture blocks	
	CW	**PT**	**CT**	**PTAG/CG**
0.5	IND	IND	PD	PD
0.25	IND	IND	PD	PD
0.2	IND	IND	PD	**ID**
0.1	IND	IND	**ID**	IND
0	**IND**	**IND**	IND	IND

IND - Imperceptible and not detectable
ID - Imperceptible and detectable
PD - Perceptible and detectable
α - Watermark Embedding Strength

III-A1. The size of P'_C pages might not be same as P_C pages. As the proposed embedding method is block wise, the mismatch in the dimension of the original cropped page P_C and the attacked cropped page P'_C will lead to incorrect detection of blocks. This may result in loss of embedded information. To avoid such scenarios, the proposed method resizes P'_C to the size of P_C using bilinear interpolation [4]. Once suspected file pages are resized, the proposed method segments P'_C pages into blocks of dimension $n \times m$ and classify them into texture and non-texture blocks using the method discussed in Section III-A1. Now, the proposed method extracts the watermark from the texture blocks using the *same watermarking method* which is used for embedding the watermark.

IV. EVALUATION CRITERIA

Unobtrusiveness and robustness are the two major requirements of any watermarking algorithm. The parameters used to assess the performance of the proposed method are discussed in this section.

A. Quality Assessment Parameters

The quality of the watermarked documents with respect to the original documents can be assessed using either subjective or objective assessment.

1) **Subjective Assessment:** In this type of assessment, readers are asked to respond to a questionnaire and rate their reading experience on a scale of 1 to 5 as shown in Table III. Although this type of assessment provides the best result in comparison with objective assessment, it is time consuming and difficult to carry out due to cultural diversities and different perception abilities of individuals. Let us assume that s subjects participated in the survey and the rating of each subject is r_{ij}, where j represents the number of subjects who participated in the survey of language i. A subjective Mean Opinion Score (MOS) can be computed as $MOS_i = \sum_{j=1}^{s} r_{ij}/s$.
2) **Objective Assessment:** In this type of assessment, mathematical tools are used to estimate the quality of the watermarked content with respect to the original content. This type of assessment can be done easily as it does not have any human involvement. The proposed method uses Peak Signal to Noise Ratio (PSNR) and the Structural Similarity Index (SSIM) to measure the quality of the watermarked documents [20]. Watermarked content with PSNR value greater than 30 dB [19] or SSIM close to 1 [20] is considered as indistinguishable from original content.

[4]We kept the size of cropped page P_C same across all the documents. We pass the size of P_C as an argument to the extractor. In case, the size of P_C and P'_C do not match, we resize P'_C to P_C using bilinear interpolation.

TABLE II. NOTATIONS USED FOR REPRESENTING IMAGE-BASED WATERMARKING ALGORITHMS

Watermarking Algorithm	**Abbreviation**
DWT-based watermarking algorithm [14]	Algo 1
SVD-based watermarking algorithm [15]	Algo 2
DCT-based watermarking algorithm [16]	Algo 3
DWT-SVD based watermarking algorithm [17]	Algo 4
DCT-DWT-SVD based watermarking algorithm [18]	Algo 5

TABLE III. QUALITY ASSESSMENT SCALE [19]

Score	5	4	3	2	1
Quality	Excellent	Good	Fair	Poor	Bad

The proposed method also uses another matrix, *Normalized Correlation Coefficient* (NC) [21], to measure the similarity between the original watermark W and the extracted watermark W_E. The value of NC lies between 0 and 1 where 1 signifies that the original and extracted watermark are identical while 0 signifies that the original and extracted watermark are different. Since the proposed method embeds the watermark in each texture block, we shall be having as many extracted watermarks as number of texture blocks. The proposed method computes the overall correlation coefficient of the document using Algorithm 2. For a given language, the correlation coefficient is an average of correlation coefficients of all the documents in that language.

Algorithm 2: Computing correlation coefficient for the entire document

Step 1: Compute correlation coefficient for each block as: $NC(i) = \frac{\sum\sum(W-\overline{W})(W_E-\overline{W_E})}{\sqrt{\sum\sum(W-\overline{W})^2\sum\sum(W_E-\overline{W_E})^2}}$, where $1 \leq i \leq b$ and $\overline{(.)}$ represents a mean of matrix element.

Step 2: Now, we sort NC in descending order as: $NC_{sort} = sortDescending(NC)$

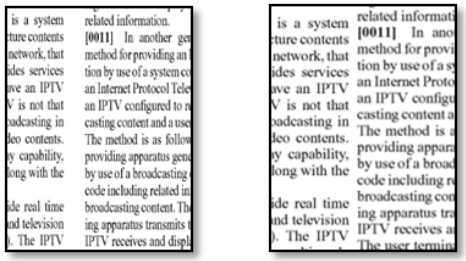
Step 3: Now, we compute the overall correlation coefficient of a document by averaging first 25% values from NC_{sort} as: $NC_{overall} = average(NC_{sort}(j)), 1 \leq j \leq 0.25b$

B. Robustness Assessment Parameters

The following attacks have been conducted to test the performance of the proposed method:

- *Print Screen:* Duplicate copies of digital documents can be easily created using "print screen" option. Print screen is a "system property" and requires expensive DRM solutions, such as CopySafe PDF [3], to disable it. In the absence of DRM solutions, print screen attack can be conducted in following ways:
  1) *Using a screen shot:* This is the easiest way of capturing the content which is currently being displayed on the screen. A malicious user can press the "Take a screen shot" option in the document file and manually select the page. This will generate an image which can be converted easily to a PDF file. These PDF files can be distributed or shared illegally. Manual print screen leads to *scaling* as well as *cropping* attacks.
  2) *Using software:* Software and Application Programming Interfaces, such as ClearestPDFToImage [22], allow a user to easily manipulate the PDF documents. PDF files at different quality levels can be generated using these softwares and APIs. To conduct this attack, we have used ClearestPDFToImage software [22]. Further, we have considered 3 quality levels for documents: (i) Low, (ii) Normal, and (iii) Good. Print screen using software leads to *scaling* attack, *compression* attack, and slight rotations in the text. Down scaling occurs when the attack is conducted at low quality while up scaling occurs when the attack is conducted at normal as well as good quality.

  We have assumed that the PDF generated through print screen attack will not be useful if the text content (text, graphs, images, tables, etc.) is cropped. Hence, we have conducted the print screen experiment while preserving maximum of the text content.
- *Print and Scan Attack:* The print and scan process has been widely studied in literature for the robustness assessment of text watermarking methods. Printing is a "document property" which can be disabled easily using PDF processing tools such as *JPedal* [23]. Though the printing option can be easily disabled using PDF processing tools, there are other ways to enable the printing option in a PDF file. For instance, a malicious user can try to enable the printing option by transcoding a PDF file (PDF to PS to PDF). In such a case, the malicious user will be able to "print and scan" the document. To study the robustness of the proposed method against "print and scan" process, we printed the documents and scanned them at different DPI values i.e. 100, 200 and 300. In our experiments, we have used Canon's black and white printer cum scanner (ImageRunner 2520).
- *Text alteration:* Malicious users can try to remove the watermark by altering the content of the text document. To check the robustness of the proposed method, we modified the content of the text document by: (i) transforming the sentence, (ii) replacing words with synonyms, (iii) highlighting the text, and (iv) manually strike through (multiple times) the word and then writing it back on top of strike through word.
- *Stitching Attack:* In the proposed method, we treat each page of the document as an image. A malicious user may try to crop the pages and stitch them back to produce an exact or similar copy. Stitching of pages may lead to automatic destruction of the embedded watermark. To study the robustness of the proposed method, we have conducted this attack in two ways:



(a) Before (b) After

Fig. 9. Example of column-wise stitching

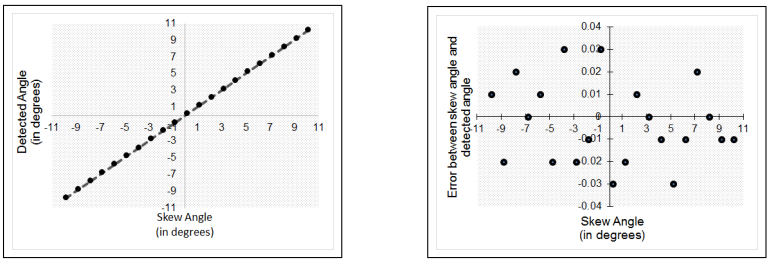
(a) Page 1 (b) Page 2 (c) After Stitching

Fig. 10. Example of page-wise stitching: (a-b) Two consecutive pages of a document (c) Resultant page obtained after stitching (a) and (b)

1) *Column/Row-wise stitching:* To conduct this attack, we cropped the pages either column or row-wise and then stitched back the cropped portions to produce nearly an exact copy. Row/Column-wise stitching leads to multiple changes in a page such as reduced space between columns, lines may be shifted up/down, and reduced space between paragraphs. Figure 9 shows a snapshot of the small portion of the page obtained by column-wise stitching. We can clearly see that column-wise stitching can lead to a change in space between columns as well shifting of lines. Such changes can lead to automatic destruction of the watermark.
2) *Page-wise stitching:* To conduct this attack, we stitched two pages to form one page, as shown in Figure 10. Page-wise stitching might not lead to a seamless reading experience. When two pages are stitched to form one page, the original pages are resized to almost half of their original width resulting in a degradation in quality of the document.

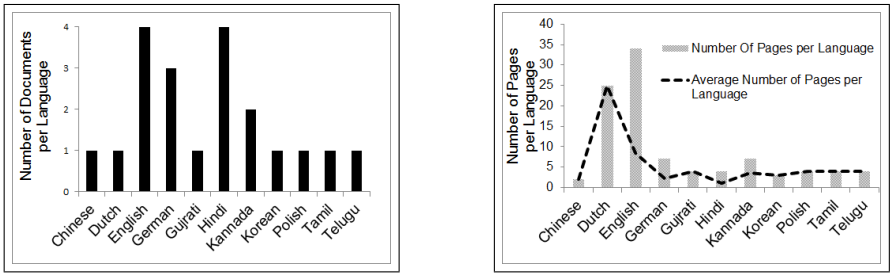
- *Signal Processing Attacks:* As the proposed method treats each document as a set of images, image-based attacks are possible on the text document. To check the robustness of the proposed method under such attacks, we have conducted following attacks:

1) *Skew or Rotation:* Skew is inevitably introduced while scanning the document. Under such circumstances, watermarking algorithms might fail to detect the watermark. To study the robustness of the proposed method under this attack, we varied the rotation angle from 0.1° to 10°. As we will see in Section V, the proposed method fails to extract the watermark in presence of skew. Since skew detection and correction is a well researched area and several robust algorithms exist in literature for detecting and correcting the skew, we integrated a well known skew detection and correction algorithm proposed by [24] in our framework. Results of skew detection are given in Figure 11. We can clearly see that the method is able to detect the skew with an error of $\pm 0.03^\circ$.
2) *Compression:* Compression is a technique to reduce the storage space required by any media. As the proposed method treats the pages of a text document as images, this attack becomes critical for analysis. Compression may lead to a drastic reduction in storage space while introducing subtle variations in the document. To conduct this attack, we have varied the JPEG quality factor from 90 to 10 where 90 denotes very low compression while 10 denotes high compression.
3) *Noise:* Generally, the watermark is considered as a noise to the signal in which it is embedded. Addition of extra noise (apart from the watermark) to the content may lead to destruction of the watermark. To check the robustness of the proposed method against such scenarios, we have added Gaussian noise G to each page of the watermarked text documents P_W to create the noisy page P_N. Noise variance σ is varied from 0.1 to 10.



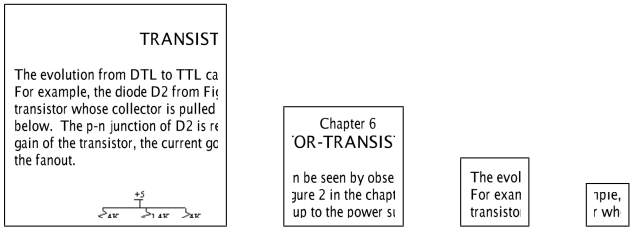
(a) (b)

Fig. 11. Results of skew detection: (a) Detected angle and (b) Error between detected angle and actual angle of rotation



(a) No. of Documents per Language (b) No. of Pages per Language

Fig. 12. Details of text document files



(a) (b) (c) (d)

Fig. 13. Texture blocks of different sizes: (a) 512 × 512, (b) 256 × 256, (c) 128 × 128, (d) 64 × 64

V. Experimental Results

A. Experimental Environment

To test the performance of the proposed method, 20 text Portable Document Files (PDF's) are used. These files differ in terms of language (English, Hindi, Tamil, Chinese, etc.), number of pages (1 to 35), content type (text, graphs, tables, images, equations, color background, etc.), and page size. The details of the text documents are given in Figure 12. We have conducted the experiments on a machine having AMD Athlon II ×2 245 2.90 GHz processor and 4 GB RAM.

B. Setting of Block Size

The proposed method divides the text document into blocks. The size of the block has a direct impact on the capacity of the watermarking algorithm as well as the amount of information in the texture blocks.

- *Impact of block size on Texture Information:* Texture blocks of different sizes are shown in Figure 13. From Figure 13, it is clear that a block of size 64×64 is able to carry much less texture information while other block sizes are able to carry a significant amount of texture information.
- *Impact of block size on Capacity:* Capacity of any algorithm represents the amount of information it can carry reliably. **Algo 1:** Capacity of Algo1 is dependent upon the DWT decomposition level. Table IV provides the details of capacity of Algo1. The order of the capacity of Algo1 is:

$$Level - 1 > Level - 2 > Level - 3 \tag{2}$$

TABLE IV. CAPACITY OF ALGO 1

Block size	Algo1		
	Level - 1	Level - 2	Level - 3
512 × 512	256 × 256	128 × 128	64 × 64
256 × 256	128 × 128	64 × 64	32 × 32
128 × 128	64 × 64	32 × 32	16 × 16
64 × 64	32 × 32	16 × 16	8 × 8

Fig. 14. Original Watermark (32×32)

We embedded the watermark in documents using Algo 1 at 3 different levels. On manual inspection of the text documents, we found that the watermark is visible in Level-1 and Level-2 but not in Level-3. Hence, we selected Level-3 as the decomposition level for Algo 1. Further, Algo 1 requires $\mathcal{O}(XY)$ operations for a block having dimension $X \times Y$ [25]. Although a block size of 512×512 has more information carrying capacity then other block sizes in Table IV, it has highest computational complexity. Block sizes of 64×64 and 128×128 have much less watermark carrying capacity while block size of 256×256 is able to carry a significant amount of information. Generally, the watermarking algorithm should have a suitable tradeoff between imperceptibility, watermark carrying capacity, and computational complexity. Considering these parameters, we selected 256×256 as the block size for Algo 1.

Algo 2: For Algo 2, the capacity is directly proportional to the block size. This is because the singular values of the watermark are embedded inside the singular values of the blocks. As the watermark is binary in nature, its singular values are small in comparison to singular values of the block and in most cases, the last 6 to 7 singular values are equal to zero. Furthermore, Algo 2 requires $\mathcal{O}(XY^2 + X^2Y)$ operations for a block size of dimension $X \times Y$ [25]. Considering the computational complexity, watermark carrying capacity, and the amount of information in the texture blocks, we selected 128×128 as block size for Algo 2.

Algo 3: For global DCT-based watermarking algorithm, the capacity is directly proportional to the block size[5]. Also, Algo 3 requires $\mathcal{O}(XY)$ operations for a block size of dimension $X \times Y$. Considering the computational complexity, watermark carrying capacity, and the amount of information in the texture blocks, we selected 128×128 as block size for Algo 3.

Algo 4 & Algo 5: Algo 4 & Algo 5 are combinations of Algo 1, Algo 2, and Algo 3. Capacity of Algo 4 and Algo 5 is equal to $\frac{1}{4}^{th}$ of the block size. However, computational complexities of Algo 4 and Algo 5 are different. On a block of dimension $X \times Y$, Algo 4 require $\mathcal{O}(XY)$ and $\mathcal{O}(XY^2 + X^2Y)$ operations for computing DWT and SVD respectively while Algo 5 require $\mathcal{O}(XY)$, $\mathcal{O}(XY)$, and $\mathcal{O}(XY^2 + X^2Y)$ operations for computing DWT, DCT, and SVD respectively. Considering computational complexity, embedding capacity, and amount of information in the texture blocks; we selected 128×128 as block size for Algo 4 and Algo 5.

For a comparative study of different image watermarking algorithms, the proposed method used a watermark of dimension 32×32, shown in Figure 14.

C. Impact on File Size

The watermark acts like a noise to the document and embedding of watermark will have an impact on the document size. As the proposed method is block-wise approach, we compared the size of the blocks before and after watermarking. We found that the proposed method increases the size of CT blocks by approximately 0.7% while the size of PTPG and CG blocks by approximately 1%. The increase in block size is quite less and hence, we can say that the proposed method is practical.

D. Quality Assessment

Embedding of a watermark inside the text document may degrade the quality of the text document. We have used the parameters discussed in Section IV-A to assess the quality of the watermarked text documents with respect to the original text documents.

1) Impact of Watermarking on Different Elements of Text Document: A text document contain different elements such as text, equations, images, graphs, and charts. Figure 15 shows the snapshots of a few elements present in the text document. From Figure 15, it is clear that watermarking of the text document with different watermarking algorithms does not deteriorate the quality of different elements present inside the text documents.

2) Subjective Assessment: Subjective assessment relates to the actual reading experience. Carrying out such reading assessments requires sufficient cultural, racial, and gender diversity of readers. Hence, we have limited our reading assessment as an informal one, limited to **24** subjects. Subjects are under-graduate students, graduate students, researchers, faculty, and employees. Among these 24 subjects, a few of them participated in multiple languages: 10 subjects participated in 2 languages, 2 subjects participated in 3 languages, and 1 subject participated in 4 languages.

[5]We have not used local DCT-based watermarking algorithm in which image is divided into a size of 8×8 and watermark is embedded inside these 8×8 blocks. The reason is quite obvious. As we saw in Figure 13, the information carrying capacity reduces with the decrease in block-size. If we reduce the block-size to 8×8, then majority of the blocks are either black or white. If we try to embed the watermark in such blocks, then watermark is either automatically destructed or becomes visible.

We shared the original as well as the watermarked documents (watermarked using different algorithms) with the subjects and asked the following questions:

1) Are there any distortions in the characters (such as discontinuities, missing links in characters, etc.) due to watermarking?
2) Is there any loss of information in graphs, images, and tables due to watermarking?
3) Are there any other visual distortions in the document due to watermarking?
4) Rate your overall reading experience.

As per the survey, subjects were not able to identify any difference between the original and watermarked documents. Based on the ratings of subjects, the average MOS for each language is computed and is shown in Figure 16(a). Average MOS across all the languages is 4.63. Average MOS values clearly indicate that watermarked documents obtained using different watermarking algorithms are of good quality and the subjects were not able to distinguish between the original and watermarked documents.

3) Objective Assessment: Figures 16(b) and 16(c) contain the PSNR as well as SSIM index values. From Figures 16(b) and 16(c), it is clear that PSNR values for all watermarking algorithms under consideration are greater than 45 dB while the SSIM index values are greater than 0.99 (close to 1). These values clearly indicate that the degradations introduced to the original document due to watermarking are minimal. This is consistent with the observations obtained from the subjective assessment.

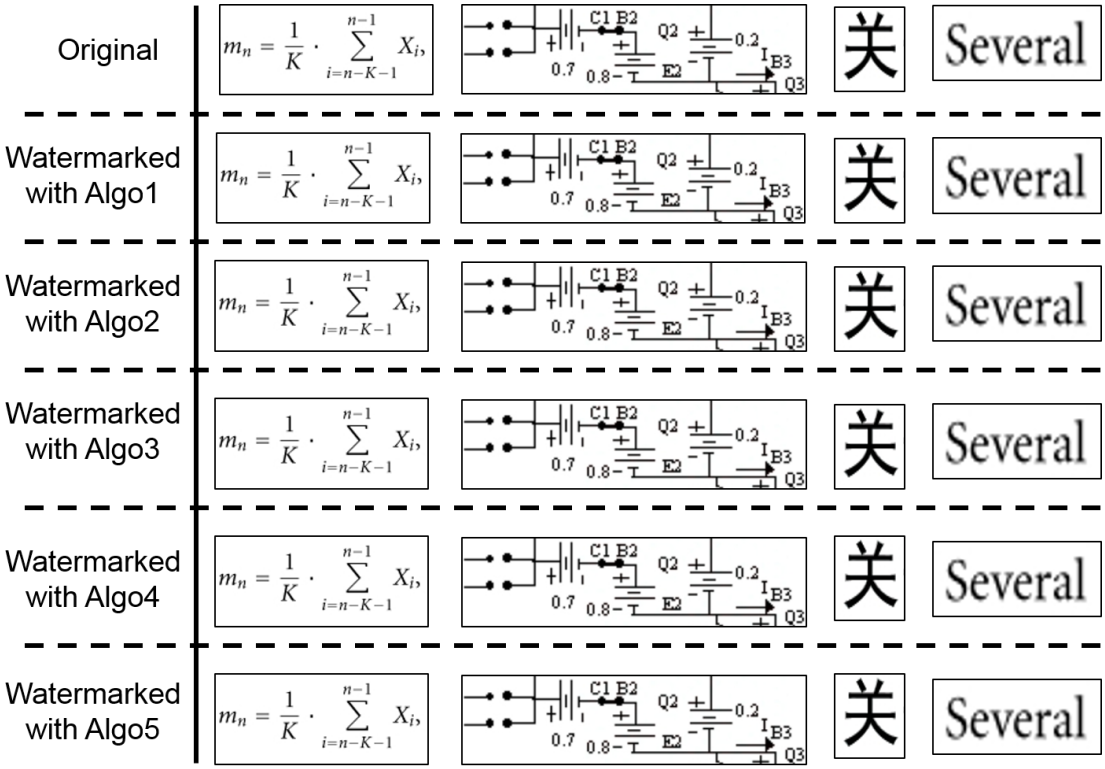
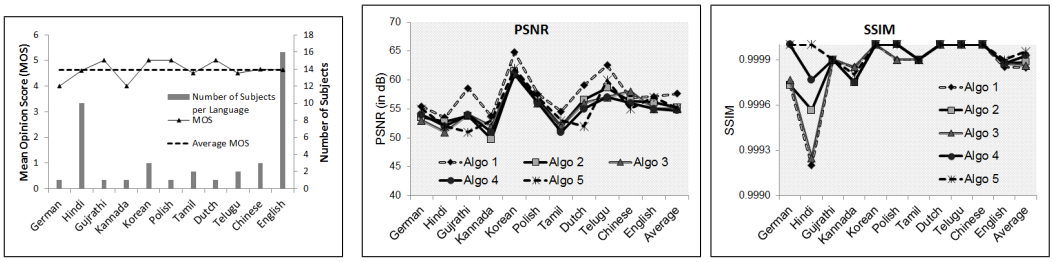


Fig. 15. Impact of Watermarking on different elements (characters, equations, etc.) of a text document



(a) Subjective Assessment (b) PSNR (c) SSIM

Fig. 16. Quality Assessment: (a) Subjective Assessment, (b-c) Objective Assessment

E. Robustness Assessment

The robustness of the proposed method under different attacks is discussed below:

1) Without Any Attack: Figure 17 shows the performance of watermarking algorithms without any attack. For all languages under consideration, the average value of correlation coefficient across different languages for Algo 1, Algo 2, Algo 3, Algo 4,and Algo 5 are 0.81, 0.96, 0.74, 0.94, and 0.86 respectively.

2) Print Screen Attack: Print screen attack can either scale or crop or slightly rotate the content. Manual print screen attack leads to scaling as well as cropping while print screen attack using software leads to scaling, compression, and slight rotations.

- *Using screen shot option*[6]*:* Figure 18 shows the performance of different algorithms under print screen attack conducted using the screen shot option. For all languages under consideration, the average value of correlation coefficient across different languages for Algo 1, Algo 2, Algo 3, Algo 4,and Algo 5 are 0.76, 0.94, 0.70, 0.86, and 0.80 respectively.
- *Using Software:* Figure 19 shows the correlation coefficient of the extracted watermark with respect to the embedded watermark. It is clear from Figure 19 that the performance of watermarking algorithm remains almost constant with change in quality of the document from low to good. The average correlation coefficient value across all languages for Algo 1, Algo 2, Algo 3, Algo 4, and Algo 5 are [0.75, 0.79], [0.94, 0.95], [0.66, 0.72], [0.86, 0.90], and [0.78, 0.84] respectively when the quality of the document is changed from low to good.

3) Print and Scan Attack: Figure 20 shows the performance of different watermarking algorithms under "print and scan" attack at different DPI values. For all the documents under study, the average value of correlation coefficient across different languages for Algo 1, Algo 2, Algo 3, Algo 4, and Algo 5 are [0.52, 0.58], [0.84, 0.86], [0.32, 0.48], [0.79, 0.85], and [0.69, 0.78] respectively. The variation in correlation coefficient is because of the variation in content. We observed that the correlation coefficient values are lower for documents in which most of the content is colored graphics. There is a significant loss of information when we printed colored documents using a black and white printer which resulted in lower correlation coefficient of the extracted watermark.

4) Text Alteration: Figure 21(b) results of text document on one of the English document. The document contains 400 texture blocks in which watermark was embedded. We modified the content of the text document by: (i) transforming the sentence, (ii) replacing words with synonyms, (iii) highlighting the text, and (iv) manually strike through (multiple times) the word and sentences. Snapshots of two attacked blocks and extracted watermarks from these blocks are shown in Figure 21(a). We noticed that the proposed method is able to detect the watermark from the blocks with minor alterations (such as strike though a single word or synonym substitution) while fails to extract the watermark from blocks with major alterations (e.g. strike though a sentence or sentence transformation).

We conducted the similar experiments on different documents and results are summarized in Figure 21(c). For all the documents under study, the average value of correlation coefficient across different languages for Algo 1, Algo 2, Algo 3, Algo 4, and Algo 5 are 0.78, 0.93, 0.69, 0.89, and 0.83 respectively. These values clearly indicate that the proposed method is able to sustain this attack.

5) Stitching Attack: Figure 22 shows the performance of different algorithms under "stitching" attack. For all documents under consideration, the average value of correlation coefficient across different languages for Algo 1, Algo 2, Algo 3, Algo 4, and Algo 5 are 0.78, 0.92, 0.68, 0.88, and 0.78 respectively. These values indicate that the algorithms are able to handle the row/column-wise stitching attack. On the other hand, the proposed method is not able to handle the page-wise stitching attack (correlation coefficient values of the extracted watermarks using different watermarking algorithms are close to zero). Due to page-wise stitching, the content inside the blocks changed and hence, the proposed method failed to extract the watermark. It is worth noting that the proposed method is able to classify the blocks into different categories such as CW, PT, and CG under stitching attack (row/column-wise as well as page-wise stitching).

6) Rotation attack: Figure 23 shows the correlation coefficient of the extracted watermark with respect to the embedded watermark at different rotation angles. As we increase the angle of rotation, the method fails to detect the watermark in absence of skew detection and correction algorithm. However, the robustness of the proposed method increases after skew detection

[6]While conducting this attack, we tried to preserve around 95% of the text content.

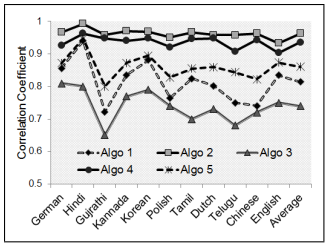


Fig. 17. Robustness without any Attack

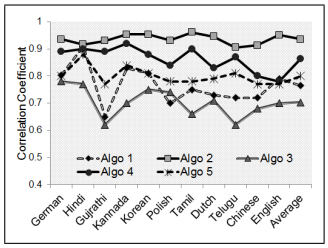
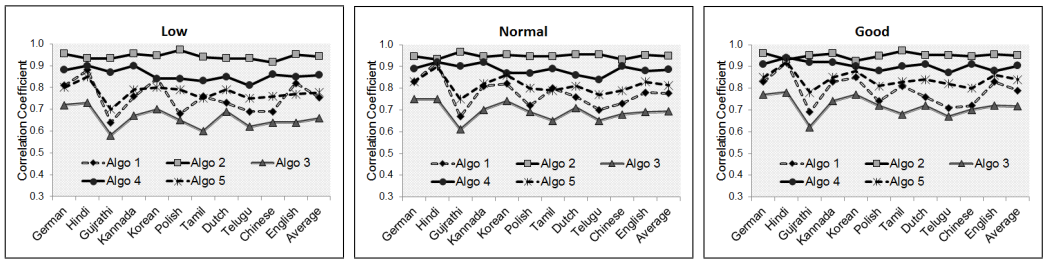
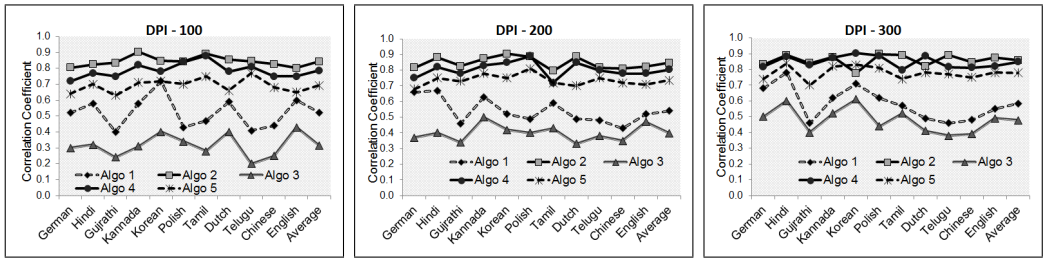


Fig. 18. Robustness against print screen attack conducted using Screen Shot



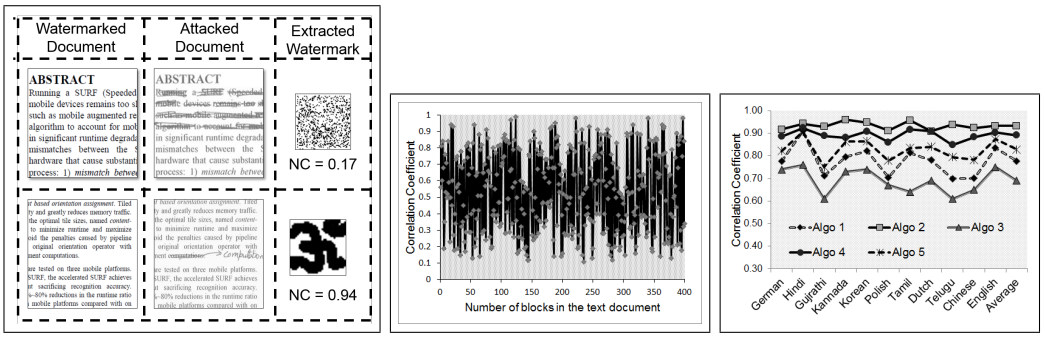
(a) Low (b) Normal (c) Good

Fig. 19. Robustness against Print Screen Attack conducted using software



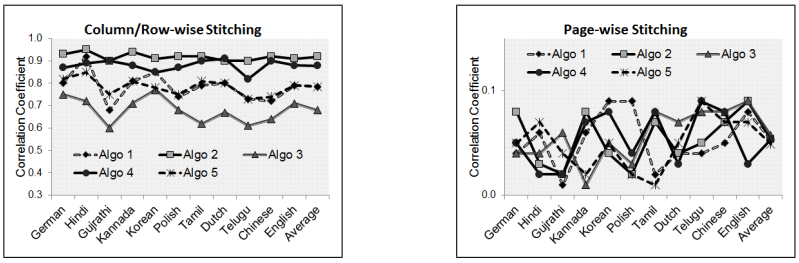
(a) DPI - 100 (b) DPI - 200 (c) DPI - 300

Fig. 20. Robustness against Print and Scan Attack at different DPI values



(a) (b) (c)

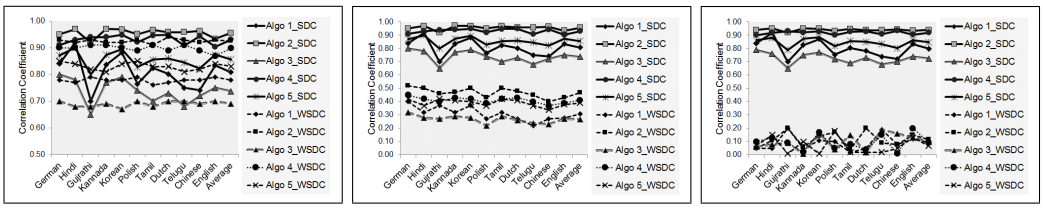
Fig. 21. Results of text modification: (a) Snapshots of attacked block and extracted watermark (using Algo 2), (b) Correlation coefficient of extracted watermarks from all blocks of a document (using Algo 2) and (c) Correlation coefficient of watermarks extracted using different watermarking algorithms across different languages



(a) Column/Row-wise Stitching (b) Page-wise Stitching

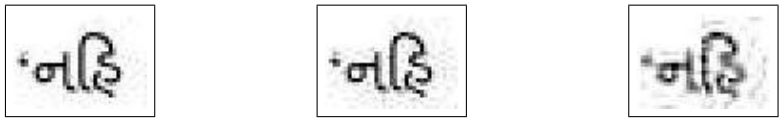
Fig. 22. Robustness against Stitching Attack

and correction. For all documents under consideration, the average value of correlation coefficient (after skew detection and correction) across different languages for Algo 1, Algo 2, Algo 3, Algo 4, and Algo 5 are between [0.80, 0.81], [0.94, 0.96],



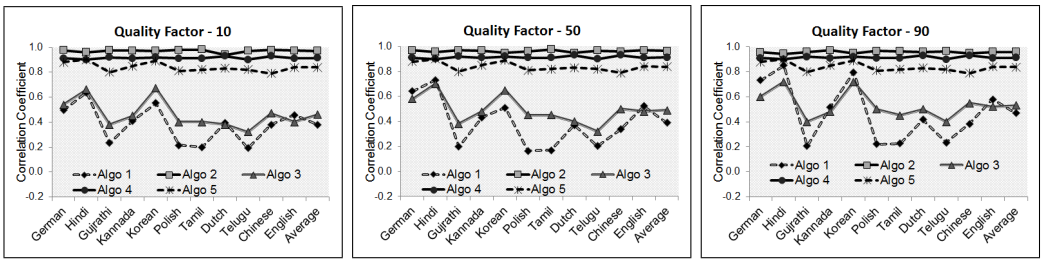
(a) Rotation = 1 ° (b) Rotation = 5 ° (c) Rotation = 10 °

Fig. 23. Robustness against Rotation Attack (SDC -with skew detection and correction; WSDC - Without skew detection and correction)



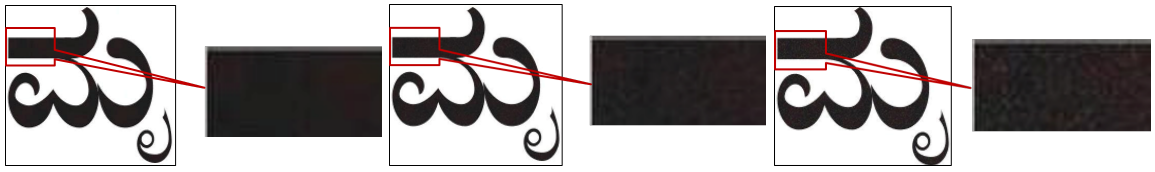
(a) Q-factor = 90 (b) Q-factor = 50 (c) Q-factor = 10

Fig. 24. Effect of compression on words and characters (this word is from the Gujrati Language)



(a) Q-10 (b) Q-50 (c) Q-90

Fig. 25. Robustness against Compression Attack



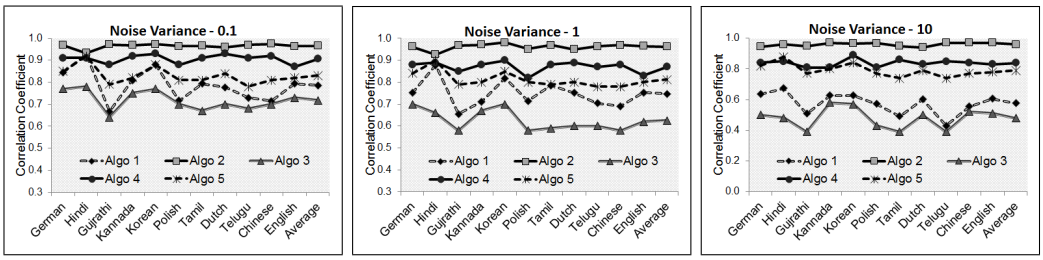
(a) variance = 0.1 (b) variance = 1 (c) variance = 10

Fig. 26. Effect of Noise on characters (character is from Kannada Language)

[0.73, 0.74], [0.92, 0.93], and [0.85, 0.86] respectively when we varied the degree of rotation from 1° to 10°.

7) Compression Attack: Figure 24 shows the effect of compression on characters. From Figure 24, it is clear that decrease in quality factor makes the words and characters blurry and hence, obscures reading. Besides blurriness, compression may destroy the embedded watermark. Figure 25 shows the correlation coefficient of the extracted watermark with respect to the embedded watermark at different quality factors. For all documents under consideration, the average value of correlation coefficient across different languages for Algo 1, Algo 2, Algo 3, Algo 4, and Algo 5 are between [0.38, 0.47], [0.96, 0.97], [0.46, 0.53], [0.90, 0.91], and [0.83, 0.84] respectively when we varied quality factor from 10 to 90.

8) Noise Attack: Figure 26 shows the effect of noise addition on characters. From Figure 26, it is clear that noise addition does not have a huge impact on the readability of characters. However, addition of noise may destroy the embedded watermark. Figure 27 shows the correlation coefficient of the extracted watermark with respect to the embedded watermark at different noise levels. For all documents under consideration, the average value of correlation coefficient across different languages for Algo 1, Algo 2, Algo 3, Algo 4, and Algo 5 are between [0.58, 0.79], [0.95, 0.96], [0.48, 0.72], [0.84, 0.91], and [0.79, 0.83] respectively when we varied noise from 0.1 to 10.



(a) variance = 0.1 (b) variance = 1 (c) variance = 10

Fig. 27. Robustness against Noise Attack

F. Discussion

The performance of different watermarking algorithms in terms of robustness, computational complexity, and imperceptibility is discussed in above sub-sections. Though all the image-based watermarking algorithms are imperceptible, the robustness against attacks and computational complexity of these algorithms is different. The order of robustness and computational complexity are given in Eqns. 3 and 4 respectively.

$$Algo2 > Algo4 > Algo5 > Algo1 > Algo3 \tag{3}$$

$$Algo3 > Algo1 > Algo4 > Algo5 > Algo2 \tag{4}$$

Algo 2 is more robust then other algorithms. This is due to inherent properties of Algo 2 such as resiliency against noise and scaling [15]. It is worth noting that when we did hybrid watermarking by combining different transform-based algorithms (DCT and DWT) with SVD, the robustness of the hybrid algorithms is increased in comparison to their native implementations. The increase in robustness is due to the inherent properties of SVD.

Another point worth noting is that correlation coefficient for few languages such as Hindi, German, and Korean is greater than other languages. This is due to the fact that the documents in these languages contain more graphical data (for instance, documents in Hindi language have a colored background) than others which increased their robustness against attacks such as compression.

In summary, we can say that Algo 2 is a better option when robustness is a prime concern while Algo 4 is better option when both computational complexity and robustness are prime concerns.

G. Application in fingerprinting

Fingerprinting aims at embedding a unique watermark for every user. Figure 28 shows an example of fingerprinting where three instances of watermarked text files are created by embedding three different watermarks. In the proposed method, we generate anti-collusion codes (or unique watermarks or fingerprints) using the method proposed by [26] and embed them using Algo 2[7]. Fingerprints are orthogonal i.e. the correlation between different fingerprints is very less or can be ignored. The generation of anti-collusion codes is out-of-the-scope of this paper and hence, we restrict ourselves from discussing it. However, we encourage readers to read the method proposed by [26].

Collusion attack is the widely studied attack against watermarking systems aimed at fingerprinting. When a user came with U watermarked copies of same content, he or she can simply average these U copies to create a colluded version of the content. Figure 29 shows the result of average collusion attack. From Figure 29, we can see that the proposed method is able to detect at least 90% of the colluders when number of colluders are less than 20. However, the detection rate drops to $\approx 10\%$ when number of colluders are increased from 20 to 60. The decrease in detection rate is obvious as the quality of colluded content decreases with increase in number of colluders (average PSNR of colluded text documents drop from 55 dB to 14 dB as we increase number of colluders from 2 to 60).

H. Comparison with related work

In Table V, we compare the proposed method with state-of-art methods. The capacity of existing watermarking methods is a variable and entirely related to content length and its type. For instance, the embedding capacity of the method proposed by [27] is proportional to number of lines in a document. In contrast to existing methods, the embedding capacity of the proposed method is dependent upon the image watermarking algorithm and block size.

Further, existing algorithms and the proposed method are robust to print & scan attack. However, existing algorithms cannot be applied for fingerprinting because malicious users can easily detect and destroy the embedded watermarks by simply comparing

[7]We have chosen the most robust algorithm, however other algorithms can also be used.

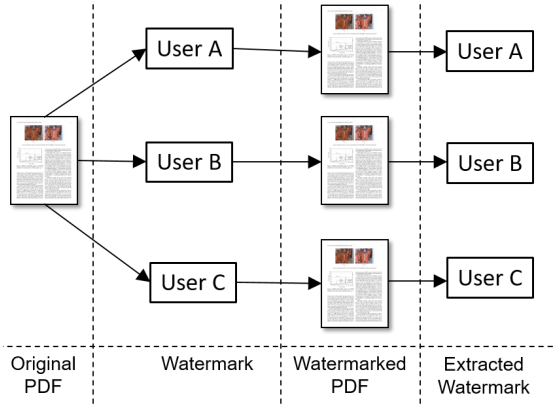


Fig. 28. Example of fingerprinting

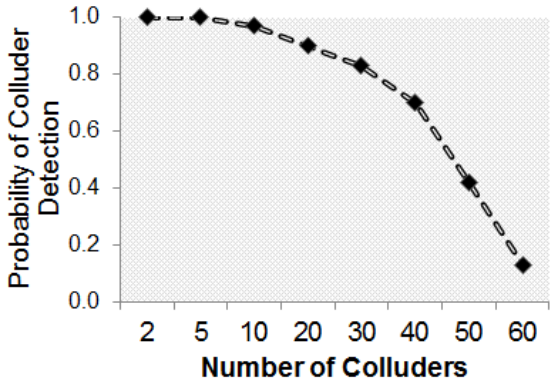


Fig. 29. Results for Collusion Attack

TABLE V. COMPARISON WITH EXISTING METHODS

Method	Capacity	Robustness		Multiple language Support
		Print & Scan	Fingerprinting	
[27]	approx. equal to no. of lines in a document	Yes	No	Yes
[28]	approx. equal to no. of words in a document	Yes	No	Yes
[5]	approx. equal to no. of special characters	Yes	No	No
[12]	equal to no. of sentence transformations	Yes	No	Yes
[10]	depends upon no. of high frequency words in a sentence	Yes	No	Yes
[13]	equal to no. of word substitutions	Yes	No	Yes
Proposed method	depends upon image watermarking algorithm	Yes	Yes	Yes

the different copies of a same content. Unlike existing methods which embeds the watermark by changing text specific features such as modulation of distance between words and characters, the proposed method uses image-based watermarking algorithm for embedding the watermark. We observed that image-based watermarking algorithms doesn't alter the text specific features and can be used for fingerprinting.

VI. CONCLUSION

This paper proposes a framework that facilitates the usage of image watermarking algorithms in text documents. The proposed method divides the document into blocks and classifies them into texture and non-texture blocks using energy. The watermark is then embedded inside the texture blocks using content adaptive watermarking strength. For evaluation, the proposed approach is integrated with known image watermarking methods. Experimental results clearly indicate that the proposed method facilitates the usage of known image watermarking algorithms in text documents and is robust attacks.

Besides this, the proposed method is language independent and does not require manual introspection like NLW methods. As the proposed method helps in adopting image watermarking algorithms in text documents, it can be used in different applications, such as fingerprinting, where existing methods cannot be used.

REFERENCES

[1] Wen Zeng and Aad van Moorsel. Quantitative evaluation of enterprise DRM technology. *Electronic Notes in Theoretical Computer Science*, 275(0):159 – 174, 2011.

[2] Jorge L. Hernandez-Ardieta, Ana I. Gonzalez-Tablas, Jose M. de Fuentes, and Benjamin Ramos. A taxonomy and survey of attacks on digital signatures. *Computers & Security*, 34(0):67 – 112, 2013.

[3] CopySafe. https://www.copysafe.net/copysafe_pdf.asp, 2015.

[4] Min Wu and B. Liu. Data hiding in binary image for authentication and annotation. *Multimedia, IEEE Transactions on*, 6(4):528–538, 2004.

[5] Zhang Wenyin and Jin Ningde. Chinese text watermarking based on occlusive components. In *Proceedings of the 2006 International Conference on Privacy, Security and Trust: Bridge the Gap Between PST Technologies and Business Services*, PST '06, pages 61:1–61:4, New York, NY, USA, 2006. ACM.

[6] P.V.K. Borges, J. Mayer, and E. Izquierdo. Robust and transparent color modulation for text data hiding. *Multimedia, IEEE Transactions on*, 10(8):1479–1489, 2008.

[7] R. Villan, S. Voloshynovskiy, O. Koval, J. Vila, E. Topak, F. Deguillaume, Y. Rytsar, and T. Pun. Text data-hiding for digital and printed documents: Theoretical and practical considerations. In *In proceedings of SPIE-IS&Electronic Imaging 2006, Security, Steganography, and Watermarking of Multimedia Contents VIII*, 2006.

[8] Ding Huang and Hong Yan. Interword distance changes represented by sine waves for watermarking text images. *Circuits and Systems for Video Technology, IEEE Transactions on*, 11(12):1237–1245, 2001.

[9] Adnan M. Alattar and Osama M. Alattar. Watermarking electronic text documents containing justified paragraphs and irregular line spacing. In *Proceedings of SPIE - Volume 5306, Security, Steganography, and Wateramaking of Multimedia Contents VI*, pages 685–695, 2004.

[10] Jixian Zhang, Qinglin Wang, Yuan Li, Dongmei Li, and Yuexing Hao. A method for chinese text classification based on three-dimensional vector space model. In *Computer Science Service System (CSSS), 2012 International Conference on*, pages 1324–1327, 2012.

[11] Wiyada Yawai and N. Hiransakolwong. Increase the hiding-bit capacity and strength for text watermarking with the line intersection on text image. In *Computing Technology and Information Management (ICCM), 2012 8th International Conference on*, volume 1, pages 427–433, 2012.

[12] Mercan Topkara, Umut Topkara, and Mikhail J. Atallah. Words are not enough: Sentence level natural language watermarking. In *Proceedings of the 4th ACM International Workshop on Contents Protection and Security*, MCPS '06, pages 37–46, New York, NY, USA, 2006. ACM.

[13] Oren Halvani, Martin Steinebach, Patrick Wolf, and Ralf Zimmermann. Natural language watermarking for german texts. In *Proceedings of the First ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '13, pages 193–202, New York, NY, USA, 2013. ACM.

[14] A. Adhipathi Reddy and B.N. Chatterji. A new wavelet based logo-watermarking scheme. *Pattern Recognition Letters*, 26(7):1019 – 1027, 2005.

[15] Ruizhen Liu and Tieniu Tan. An SVD-based watermarking scheme for protecting rightful ownership. *Multimedia, IEEE Transactions on*, 4(1):121–128, 2002.

[16] Ingemar J. Cox, Joe Kilian, F.T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. *Image Processing, IEEE Transactions on*, 6(12):1673–1687, 1997.

[17] Emir Ganic and Ahmet M. Eskicioglu. Robust dwt-svd domain image watermarking: Embedding data in all frequencies. In *Proceedings of the 2004 Workshop on Multimedia and Security*, MM&Sec '04, pages 166–174, New York, NY, USA, 2004. ACM.

[18] K.A. Navas, M.C. Ajay, M. Lekshmi, T.S. Archana, and M. Sasikumar. Dwt-dct-svd based watermarking. In *Communication Systems Software and Middleware and Workshops, 2008. COMSWARE 2008. 3rd International Conference on*, pages 271–274, Jan 2008.

[19] Nader H. H. Aldeeb, Ibrahim S. I. Abuhaiba,"An Improved Watermarking Scheme for Tiny Tamper Detection of Color Images", I.J. Image, Graphics and Signal Processing, 2013, 5, 1-14.

[20] S. Chikkerur, V. Sundaram, M. Reisslein, and L.J. Karam. Objective video quality assessment methods: A classification, review, and performance comparison. *Broadcasting, IEEE Transactions on*, 57(2):165–182, 2011.

[21] Ching-Yung Lin, Min Wu, J.A Bloom, Ingemar J. Cox, M.L. Miller, and Yui Man Lui. Rotation, scale, and translation resilient watermarking for images. *Image Processing, IEEE Transactions on*, 10(5):767–782, May 2001.

[22] Clearest. http://www.clarest.com/site/page008.aspx, 2015.

[23] IDRSolutions. https://www.idrsolutions.com/, 2015.

[24] Yang Cao, Shuhua Wang, and Heng Li. Skew detection and correction in document images based on straight-line fitting. *Pattern Recognition Letters*, 24(12):1871–1879, 2003.

[25] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996.

[26] W. Trappe, Min Wu, Z.J. Wang, and K.J.R. Liu. Anti-collusion fingerprinting for multimedia. *Signal Processing, IEEE Transactions on*, 51(4):1069–1087, Apr 2003.

[27] S.H. Low, N.F. Maxemchuk, J.T. Brassil, and L. O'Gorman. Document marking and identification using both line and word shifting. In *INFOCOM '95. Fourteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Bringing Information to People. Proceedings. IEEE*, pages 853–860, Apr 1995.

[28] Young-Won Kim, Kyung-Ae Moon, and Il-Seok Oh. A text watermarking algorithm based on word classification and inter-word space statistics. In *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on*, pages 775–779, Aug 2003.

[29] M. Barni, F. Bartolini, and A. Piva. Improved wavelet-based watermarking through pixel-wise masking. *Image Processing, IEEE Transactions on*, 10(5):783–791, May 2001.

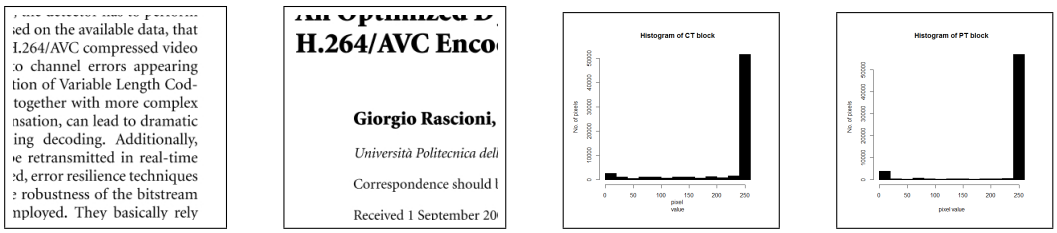
VII. Appendix

A. Histogram-based block classification method

Histogram captures the information about the distribution of data inside the image. It can be a powerful tool to classify the blocks into two categories i.e. text and graphics. Unlike the energy based approach, it cannot be used for fine grained classification of blocks such as classifying the blocks into CT and PT category. This is because blocks such as CT/PT and CG/PG/PTPG exhibits similar histogram properties and may cause misclassification of the blocks. Figure 30 shows the histogram of CT and PT blocks. It is evident from the figure that both the blocks exhibit similar histogram properties and hence, fine-grained classification of such blocks is very difficult.

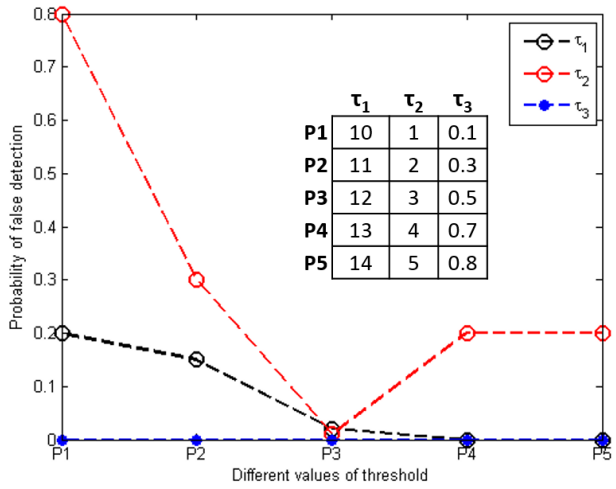
Block classification using histogram is elaborated in Algorithm 3. The performance of Algorithm 3 is dependent upon thresholds (τ_1, τ_2, and τ_3). We conducted several experiments to adjust the value of τ_1, τ_2, and τ_3. Figure 31 shows the relationship between the probability of false detection and thresholds. From Figure 31, we can clearly see that probability of false detection is less when thresholds for block classification using histogram (τ_1, τ_2, and τ_3) are 12, 3, and 0.5 respectively. Figure 32 shows the results for one of the documents used in our experiments. The document contains 88 blocks of dimension 256×256. From Figure 32, it is clear that the histogram based method is able to classify the blocks into text and graphics category.

Apart from energy-based or histogram-based approaches, another possible approach for classification of blocks could be using mean μ_b and variance σ_b of the blocks. As the pixels in text documents take values from a few possibilities and mostly contain



(a) CT (b) PT (c) CT (d) PT

Fig. 30. Histogram of different blocks in a text document: (a-b) Different blocks and (c-d) Histogram of blocks



	τ_1	τ_2	τ_3
P1	10	1	0.1
P2	11	2	0.3
P3	12	3	0.5
P4	13	4	0.7
P5	14	5	0.8

Fig. 31. Relationship between block classification and thresholds

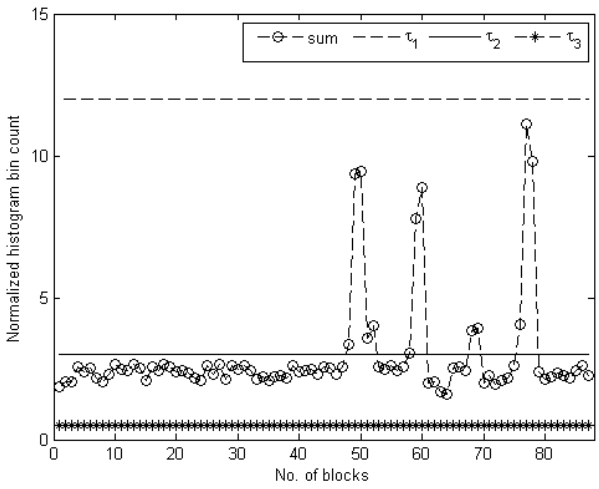


Fig. 32. Example: Histogram based block classification

TABLE VI. MEAN μ_b AND VARIANCE σ_b OF DIFFERENT BLOCKS IN TEXT DOCUMENTS

Block Type	CT	PT	PTPG	CG
μ_b	232.05	226.95	223.63	224.4
σ_b	12.49	19.89	18.87	14.53

black and white pixels, it will be very difficult to classify the blocks using mean μ_b and variance σ_b, as shown in Table VI. Hence, we haven't explored this approach.

Algorithm 3: Histogram-based Classification of Blocks

Read the block and compute its histogram H.

Normalize the histogram using L2-normalization as:

$$H_{norm} = \frac{H}{\sqrt{||H||^2 + \epsilon^2}}$$

Now, compute the sum of the frequency of colors in H_{norm} as:

$$sum = \sum H_{norm}$$

if $sum > \tau_1$ *or* $sum < \tau_3$ **then**
  Non-texture block
else
  if $sum \leq \tau_1$ *and* $sum > \tau_2$ **then**
    Graphics block
  else if $sum \leq \tau_2$ *and* $sum > \tau_3$ **then**
    Text block

where τ_1, τ_2, and τ_3 are thresholds used for block classification using histogram.

B. Brief overview of image watermarking algorithms

In this section, we briefly describe the image watermarking algorithms.

1) DWT-based watermarking: In this subsection, DWT-based watermarking algorithm is discussed briefly. A more detailed explanation about the algorithm can be found in [14].

Watermark Embedding - To embed a watermark W inside an image C, image C is decomposed into l levels (C_l^θ) while watermark W is decomposed into 1 level, i.e. W_0^θ, using DWT. Here, $l = \{0, 1, 2, 3\}$ represents decomposition level and $\theta \epsilon \{HH, HL, LH, LL\}$ represents the orientation. Watermark is then embedded as:

Step 1: First, find the weight factors w_l^θ for wavelet coefficients using the method given by [29].

Step 2: Now, add the watermark as: $C_l^{'\theta} = C_l^\theta + \beta_a\ w_l^\theta\ W_0^\theta$. Here, β_a represents the watermark embedding strength.

Step 3: Watermarked image C' is then obtained by taking the inverse DWT (IDWT) of $C_l^{'\theta}$.

Watermark Extraction - To extract the watermark from the possibly attacked image C'', following steps are performed:

Step 1: Watermarked image C'' and original image C are decomposed into l levels ($C_l^{''\theta}$) using DWT and then the band coefficients of watermark W' are extracted as:

$$W_0^{'\theta} = \frac{(C_{l'}^{''\theta} - C_{l'}^\theta)}{w_l^\theta} \tag{5}$$

Step 2: Now, extracted watermark band coefficients are combined with the distortion parameter as:

$$W_0^{''\theta} = W_0^{''\theta} + W_0^{'\theta} \times \left[\frac{w_l^\theta\ 2^l}{\sqrt{D_l^\theta}}\right]^2 \tag{6}$$

where D_l^θ is the distortion calculated in the neighbourhood $N_x \times N_y$ as:

$$D_l^\theta(i,j) = \frac{\sum_{x=i-\frac{N_x}{2}}^{i+\frac{N_x}{2}} \sum_{y=j-\frac{N_y}{2}}^{j+\frac{N_y}{2}} \left[(C_{l'}^{''\theta}(x,y) - C_{l'}^\theta(x,y))\right]^2}{N_x \times N_y} \tag{7}$$

Step 3: Now, extracted watermark band coefficients are normalized as:

$$W_0^{''\theta} = \frac{W_0^{''\theta}}{sum}, where\ sum = sum + \left[\frac{w_l^\theta\ 2^l}{\sqrt{D_l^\theta}}\right]^2 \tag{8}$$

Step 4: IDWT of $W_0^{''\theta}$ is taken to form the extracted watermark W''.

2) SVD based watermarking: In this subsection, SVD based watermarking algorithm is discussed briefly. A more detailed explanation about the algorithm can be found in [15].

Watermark Embedding: To embed the watermark W inside the image C, we follow below mentioned steps:

Step 1: Compute SVD of both C and W as: $C = U_c\ S_c\ V_c^T, \quad W = U_w\ S_w\ V_w^T$.

Step 2: To embed the watermark inside the cover image, modify the singular values of C as: $S_W = S_c + (\beta_b\ S_w)$. Here β_b is the watermark embedding strength.

Step 3: Now, watermarked image C' can be obtained as: $C' = U_c\ S_W\ V_c^T$.

Watermark Extraction: Following steps are performed to extract the watermark from the possibly attacked or watermarked image C'':

Step 1: Compute SVD of C'' as: $C'' = U_a S_a V_a^T$.

Step 2: Now, extract the singular values of embedded watermark as: $S'_w = \frac{(S_a - S_c)}{\beta_b}$.

Step 3: Extracted watermark W' can then be formed as: $W' = U_w S'_w V_w^T$.

3) DCT based Watermarking: In this subsection, DCT based watermarking algorithm is discussed briefly. A more detailed explanation about the algorithm can be found in [16].

Watermark Embedding: To embed the watermark W inside the image C, we follow below mentioned steps:

Step 1: Apply forward DCT to image C.

Step 2: Embed the watermark W inside the AC coefficients as: $v'_i = v_i(1 + \beta_c\ w_i)$, where v_i represents the AC coefficients obtained by applying DCT to image C and w_i is the binary representation of watermark W (During binary conversion, we represent pixel value of 255 as 1 while pixel value of 0 as 0).

Step 3: Now, apply inverse DCT on the modified coefficients to obtain the watermarked image C'.

Watermark Extraction: To extract the watermark from the watermarked or possibly attacked image C'', we follow below mentioned steps:

Step 1: Apply forward DCT to C''.

Step 2: Extract the binary watermark w'' as: $w_i'' = \frac{|v_i'' - v_i|}{\beta_c}$

Step 3: Now, create a watermark image W'' from binary watermark w'' as:

$$W_i'' = \begin{cases} 255, & if\ |w_i''| > tolerance \\ 0, & Otherwise \end{cases} \tag{9}$$

4) DWT-SVD based Watermarking: In this subsection, DWT-SVD based watermarking algorithm is discussed briefly. A more detailed explanation about the algorithm can be found in [17].

Watermark Embedding: We follow following steps for embedding the watermark:

Step 1: Compute l-level and 1-level DWT of image C and watermark W i.e. C_l^θ and W_0^θ respectively. Here, $l = \{0, 1\}$ represents decomposition level and $\theta \epsilon \{HH, HL, LH, LL\}$ represents the orientation.

Step 2: Apply SVD to each sub-band of C_l^θ and W_0^θ as: $C_l^\theta = U_{cl}^\theta S_{cl}^\theta V_{cl}^{T^\theta}, W_0^\theta = U_w^\theta S_w^\theta V_w^{T^\theta}$.

Step 3: To embed the watermark, modify the singular values of C_l^θ as: $S_{cl}^{'\theta} = S_{cl}^\theta + \beta_d S_w^\theta$. Here $\theta \epsilon \{HH, HL, LH, LL\}$ represents orientation and β_d represents the watermark embedding strength.

Step 4: Now, compute the inverse SVD to obtain the modified sub-bands as: $C_l^{'\theta} = U_{cl}^\theta S_{cl}^{'\theta} V_{cl}^{T^\theta}$.

Step 5: Apply inverse DWT to modified sub-bands $C_l^{'\theta}$ to obtain the watermarked image C'.

Watermark Extraction - Watermark is extracted from the possibly attacked or watermarked image as:

Step 1: Compute l-level DWT of image C'' i.e. $C_l^{''\theta}$.

Step 2: Compute SVD of each sub-band as: $C_l^{''\theta} = U_{cl}^{''\theta} S_{cl}^{''\theta} V_{cl}^{''T^\theta}$.

Step 3: Now, extract the sub-bands of embedded watermark as: $S_w^{'\theta} = \frac{(S_{cl}^{''\theta} - S_{cl}^\theta)}{\beta_d}$.

Step 4: Apply inverse SVD on extracted sub-bands of watermark as: $W_0^{'\theta} = U_w^\theta S_w^{'\theta} V_w^{T^\theta}$.

Step 5: Inverse DWT is now applied to these sub-bands $W_0^{'\theta}$ to obtain the watermark W'.

5) DWT-DCT-SVD based Watermarking: In this subsection, DWT-DCT-SVD based watermarking algorithm is discussed briefly. A more detailed explanation about the algorithm can be found in [18].

Watermark Embedding We follow following steps for embedding the watermark:

Step 1: Compute l-level and 1-level DWT of image C and watermark W i.e. C_l^θ and W_0^θ respectively. Here, $l = \{0, 1\}$ represents decomposition level and $\theta \epsilon \{HH, HL, LH, LL\}$ represents the orientation.

Step 2: Apply DCT to the subbands as:

$$C_{dl}^\theta = dct(C_l^\theta) \tag{10}$$

Step 3: Now, apply SVD to each sub-band of C_{dl}^θ and W_0^θ as:

$$C_{dl}^\theta = U_{cl}^\theta S_{cl}^\theta V_{cl}^{T^\theta}, W_0^\theta = U_w^\theta S_w^\theta V_w^{T^\theta} \tag{11}$$

Step 4: To embed the watermark, modify the singular values of C_{dl}^θ as

$$S_{cl}^{'\theta} = S_{cl}^\theta + \beta_e S_w^\theta \tag{12}$$

where $\theta \epsilon \{HH, HL, LH, LL\}$ represents orientation and β_e represents the watermark embedding strength.

Step 5: Now, compute the inverse SVD and inverse DCT to obtain the modified sub-bands as:

$$C_l^{'\theta} = idct(C_{dl}^{'\theta}) = idct(U_{cl}^\theta S_{cl}^{'\theta} V_{cl}^{T^\theta}) \tag{13}$$

Step 6: Now, apply inverse DWT to modified sub-bands $C_l^{'\theta}$ to obtain the watermarked image C'.

Watermark Extraction - Watermark is extracted from the possibly attacked or watermarked image as:

Step 1: Compute l-level DWT of image C'' i.e. $C_l^{''\theta}$.

Step 2: Compute DCT of each sub-band as:

$$C_{dl}^{''\theta} = dct(C_l^{''\theta}) \tag{14}$$

Step 3: Compute SVD of each sub-band as:

$$C_{dl}^{''\theta} = U_{cl}^{''\theta} S_{cl}^{''\theta} V_{cl}^{''T^\theta} \tag{15}$$

Step 4: Now, extract the sub-bands of embedded watermark as:

$$S_w^{'\theta} = \frac{(S_{cl}^{''\theta} - S_{cl}^\theta)}{\beta_e} \tag{16}$$

Step 5: Apply inverse SVD on extracted sub-bands of watermark as:

$$W_0^{'\theta} = U_w^\theta S_w^{'\theta} V_w^{T^\theta} \tag{17}$$

Step 6: Inverse DWT is now applied to these sub-bands $W_0^{'\theta}$ to obtain the watermark W'.