"Revolutionary Methodology for Detecting and Eliminating Strike-through Text in Handwritten Kannada Documents"

Mr. Sangamesh S K
Assistant Professor
Department of Computer Science and Engineering
Rural Engineering College, Hulkoti

Miss. D Sathvika Reddy
Student
Department of Computer Science And Engineering
Rural Engineering College, Hulkoti

Mrs. Pavitra M Gadar
Assistant professor
Department of Computer and Engineering
K.L.E. Institute of Technology,Hubballi

ABSTRACT:

The Document image processing is a niche field within image processing, dedicated to analyzing, enhancing, and comprehending images containing textual and graphical content from documents. This paper introduces an innovative method aimed at processing and detecting Kannada text amidst struck-out words or phrases in a database comprising handwritten Kannada data. Conventional OCR-based systems often struggle to identify struck-out text accurately, underscoring the need for alternative approaches. The dataset is divided into two categories: struck-out and non-struck-out, with a Support Vector Machine (SVM) classifier utilized for feature extraction in pattern classification. In the graph-based approach, we employ the shortest path algorithm to analyze strokes. To restore text from the input, an inpainting cleaning technique is employed. The proposed model undergoes extensive evaluation on both trained and untrained Kannada script databases.

Keywords: Strike-Out Text processing, Pattern Recognition, Handwritten OCR

1. INTRODUCTION

This research delves into the analysis of strike-out text in the Kannada script, an ancient abugida script belonging to the Brahmi family and one of the four Dravidian languages predominantly spoken in South India, particularly in Karnataka. The study proposes the utilization of a deep neural network technique for detecting strike-out words in handwritten Kannada documents. By leveraging this technique, the goal is to heighten the precision of strike-out word detection, thereby enhancing the overall comprehension and interpretation of handwritten texts in digital document analysis.The primary objective of the paper is to preprocess handwritten text extracted from an optical character recognition (OCR) system and identify strike-out strokes within Kannada literature. Illustrated in Figure 1 are various structures of strike-out strokes, encompassing single, multiple, slanted, crossed, zigzag, and wavy strokes. The frequently encountered strike-out stroke structures are as follows:Single Stroke: A single horizontal line is drawn over the word, representing the strikeout stroke.

1. Multiple Strokes: Multiple horizontal lines are drawn over the word to indicate strikeout strokes.
2. Slanted Stroke: A slanted line is used as the strikeout stroke. It can be a single slant line or multiple slant lines drawn on the word. The angle of the slant line typically ranges from 60 degrees to 10 degrees. The slant line may extend above and below the word.
3. Crossed Stroke: A crossed stroke consists of intersecting lines drawn over the word, forming an "X" shape.
4. Zigzag Stroke: Instead of a straight strikeout line, a zigzag line is used to assist in striking out the word. The zigzag line may also have a wavy nature, adding variation to the strikeout stroke. This technique helps partially hide the written word.

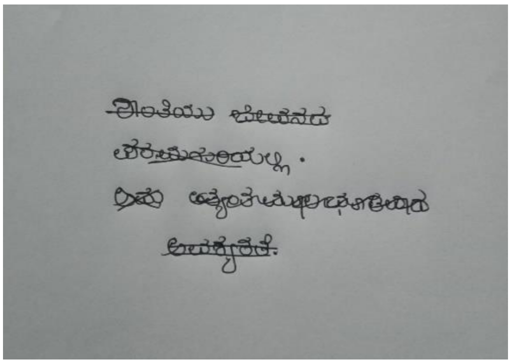


Figure 1: Various forms of strike-out strokes

The presence of these different strike-out stroke structures in handwritten Kannada literature poses a challenge for accurate strikeout detection.

In Figure 2 , a complete paragraph of text is shown with clear Strike-out Stroke(SS) of complete paragraph and the page. The analysis of handwriting reveals that consecutive words, entire paragraphs, or even entire pages can be struck out. To address this challenge, the paper proposes a comprehensive approach to categorize the handwriting script from Kannada alphabets, as well as identify struck out characters and words in the written text.

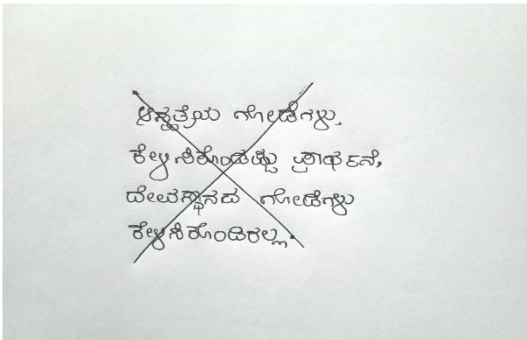


Figure 2: The SSs of complete paragraph and the page

The complete task can be divided into three subtasks:

- Detection of Normal and Strikeout Text: This subtask involves identifying and separating the normal text from the strikeout text. The goal is to distinguish between the portions of the text that are crossed out or marked for deletion and the portions that are not.

- Stroke Identification: In this subtask, the focus is on identifying and characterizing the strikeout strokes. Different types of strikeout strokes, such as single strokes, multiple strokes, slanted strokes, crossed strokes, zigzag strokes, and wavy strokes, need to be detected and classified.

- Stroke Deletion for OCR Analysis: The final subtask is to delete the strikeout strokes and generate a clean representation of the text .

By addressing these three subtasks, the proposed approach aims to provide a comprehensive solution for categorizing handwriting scripts, distinguishing strikeout text, and identifying strikeout characters and words in both Kannada . SVM is a widely used supervised learning algorithm for classification and regression.
The paper is structured as follows: In Section 2, we provide a comprehensive review of existing literature. Moving on to Section 3, the proposed methodology is discussed in detail. In Section 4, the obtained results are thoroughly examined and presented.

2. PROPOSED METHADOLOGY

Our proposed methodology for processing Kannada handwritten documents encompasses a series of distinct tasks. These include line identification, connected component analysis, recognition of strike-out text, employment of Support Vector Machine (SVM) for text classification, identification of strike-out strokes, and application of inpainting for strike-out deletion. Let's delve into each step in detail.
The initial stage of our approach entails identifying individual lines within the handwritten Kannada document. This involves segmenting the document into discrete lines, facilitating subsequent analysis.
Following line identification, we progress to identifying connected components within regions of strike-out text. Our focus is on pinpointing areas where text has been struck out, and subsequently, we isolate the connected components within these regions for further examination.
To enhance document clarity, we omit diminutive connected components that may lack substantial textual content. This step streamlines subsequent analysis by eliminating extraneous noise. Once minor components are filtered out, we isolate and separate prominent connected components. This separation aids in distinguishing

between the primary body of text and possible annotations or comments.

A pivotal task involves categorizing text segments into two classes: standard Kannada text and strike-out text. For this classification, we employ a trained Support Vector Machine (SVM) model, capable of distinguishing between these two types based on distinct features or patterns.

Within the realm of strike-out text, our focus shifts to identifying individual strike-out strokes. This step entails detecting distinct lines or strokes that constitute the strike-out annotations.

In this stage, we apply inpainting techniques to intelligently substitute strike-out annotations with appropriate content, effectively "filling in" the struck-out regions with coherent text that aligns with the surrounding context.

The workflow of this paper is represented in Figure 4.

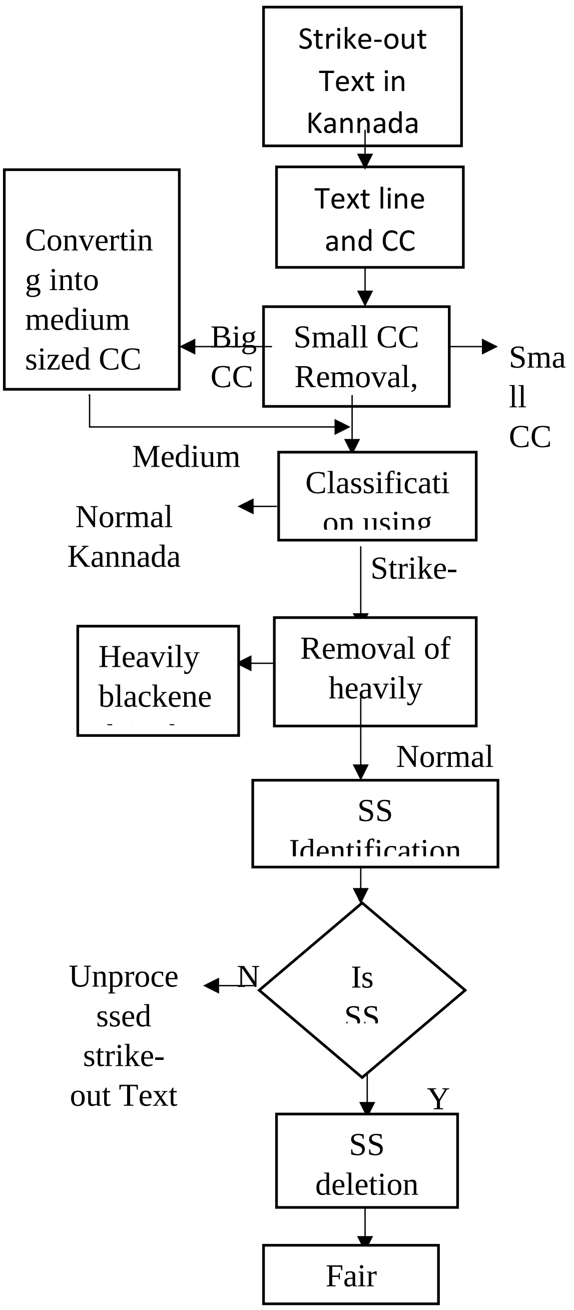


Figure 4: Workflow of Proposed Approach

In accordance with the introduction (Section 1), our study delves into the analysis of three primary forms of strike-outs commonly encountered in handwritten manuscripts: case(a) strike-outs encompassing a single word, case (b) strike-outs spanning multiple successive words, and case(c) strike-outs spanning multiple consecutive lines. The process is initiated by employing a pre-existing algorithm to identify individual lines of text within the handwritten Kannada document. This algorithm is adept at handling situations where strike-out text are present. Subsequently, we isolate and analyze the Connected Components (CCs) within each identified line. Minuscule elements like dots, dashes, commas, colons, and noise are excluded from subsequent processing stages.

Acknowledging that struck-out content usually constitutes a small fraction of the total word count, we leverage the average height (Hav) of Bounding Boxes (BBs) encompassing these components. Hav serves as a dependable gauge of the typical word height. Additionally, we calculate the standard deviation (Hsd) of BB heights for these components.

CCs meeting the criterion of BB lengths within the range of Hav to αHav and heights within Hav ± Hsd are identified as instances of type (a), as defined previously. These instances then undergo scrutiny through the Support Vector Machine (SVM) for analysis.

In our approach, we utilize a multiplier denoted as α (where $\alpha > 1$), and for our specific case, α is set to 8. This choice of α has been determined through empirical observations, particularly in the context of Kannada scripts.

When applying our method, we have found that using $\alpha = 8$ yields more favorable results. This is primarily due to the fact that most words in Kannada scripts tend to be shorter than 9 characters/ortho-syllables. However, it's important to note that if a continuous character sequence (CC) surpasses this particular length threshold, it falls into a distinct category referred to as case (b).

In a broader sense, both case (b) and case (c) are characterized by larger CCs. These larger CCs are subsequently transformed into medium-sized CCs through the process of selectively removing strokes. Once this transformation is applied, the resulting medium-sized CCs are then input into an SVM classifier.

normal components, which are not subjected to strike-outs, and strike-out components that have undergone alteration. To extract the features necessary for the SVM classifier, each continuous character sequence (CC) undergoes a process known as skeletonization. This procedure serves multiple purposes.

Firstly, it aids in identifying those components among the CCs classified by the SVM as belonging to the class that have been subjected to numerous strike-outs, rendering them irrecoverable for restoration into their original form. In instances where a word has been heavily marked, the number of iterations needed is higher compared to a normal, unmarked word. Furthermore, in such cases, the density of black pixels per unit area within the bounding box (BB) of the component exceeds that of a normal word. These distinctive traits allow us to identify heavily

3. LITERATURE REVIEW

The recognition of strike-out text in handwritten documents holds significant importance in the field of digital document analysis, enabling enhanced transcription and interpretation of handwritten content. This section presents a comprehensive review of existing literature on the recognition of strike-out text in Kannada handwritten documents, encompassing the characteristics of Kannada letters, various types of strike-outs, and the challenges associated with their detection.

Kannada, a Dravidian language, employs a script comprising 49 letters, including syllable consonants, vowels, and consonants. Each character possesses distinct features, strokes, and structures that contribute to the unique nature of the script. The complex structure of Kannada letters poses challenges in accurately identifying and analyzing strike-out text within handwritten documents.

The literature reveals a variety of strike-out techniques used in Kannada literature, aimed at crossing out or partially obscuring written content. These include single strikes, multiple strikes, slanted strikes, crossed strikes, zigzag strikes, and wavy strikes. Each type of strike-out stroke involves specific patterns and characteristics that require sophisticated recognition techniques for accurate detection.

Detecting strike-out text in Kannada handwritten documents presents several challenges due to the intricate nature of the script and the diversity of strike-out patterns. One significant challenge lies in differentiating strike-out strokes from legitimate strokes that form part of the script itself. Additionally, the presence of multiple types of strike-outs within a single document further complicates the recognition process. The varying angles, lengths, and curvatures of strike-out strokes demand robust algorithms capable of discerning subtle differences.

Moreover, the potential interference of other handwriting elements, such as underlining or marginal annotations, adds another layer of complexity. The detection of strike-out text must also account for variations in writing styles, ink densities, and paper textures, which impact stroke appearance and contrast. The challenge of accurately segmenting strike-out strokes from surrounding content is also noteworthy, particularly when dealing with overlapping or closely spaced characters.

In a related work [14], a widely adopted method for handwritten script recognition is presented. The author categorizes handwriting recognition systems into two overarching types: those based on visual appearance and those based on structural characteristics. Additionally, [15] offers a concise overview of the diverse scripts and categories employed in their methodology. The author proposes a model for comprehending and identifying text in videos and online content. It is worth highlighting that the domain of handwritten document analysis is continually evolving, necessitating further advancements to address its unique attributes, as emphasized by the author.

[15] commences by delving into existing optical character recognition (OCR) systems and their correlation with symbol identities and individual characters. The functionality of the script recognizer is elucidated with a focus on Urdu and English languages. Furthermore, the paper explores various languages utilizing the Brahmi script. The study also touches upon the prevalent script recognition methodologies employed within machine learning environments. The Spitz method, employed in this study, is elucidated through a clear block diagram, highlighting its capability to recognize multiple languages. The system achieved successful recognition of languages such as English, French, German, as well as languages with distinct shapes and varying optical densities like Chinese, Japanese, and Korean. Language recognition was accomplished based on pen position, with samples collected at a rate of 132 samples per second. The algorithm also accounted for stroke patterns and similarities. To enhance image quality, a Gaussian filter was applied. The X-axis projection of the text was utilized for analyzing word segmentation. However, it's important to note that this segmentation approach might encounter challenges when dealing with different regions of handwritten text containing diverse scripts. The Daubechies filter to establish a wavelet domain for multi-resolution analysis of handwritten numeral images. Their multi-stage recognition scheme involved cascaded Multilayer Perceptrons (MLPs). The first stage employed three MLPs for initial image classification, with only successful classifications advancing to the next stage. The second stage utilized another MLP for further classification. Accuracy evaluation using a dataset of 20 classes of handwritten Devanagari data yielded an accuracy of 70.85%. For instances of English mixed with Devanagari script, classification accuracy reached 65.02%. [17] focuses on statistical attributes like pixel density and zero crossing vertex points to differentiate Devanagari characters. Special attention is given to vertical bars and upper modifier boxes. Leverage density, moment features, and a connectionist architecture with multiple classifiers to achieve an accuracy of 89.68% in identifying Devanagari numerals. The paper also explores the use of hidden Markov models for offline word recognition, specifically addressing challenges posed by superimposed strokes without additional identification or cleaning.

3.1 Strike-out Stroke as Shortest Path

In the earlier step, it is considered to have a reasonable straight SS in the image. In the graph for, the shortest path between left and right node indicates the reasonable straight strike out. In this paper, the approach is based on 2 main observations. The first one is to Observe and horizontal line which covers the width of the component. The second one is to find reasonably straight and continuous line which is unbroken in nature.

3.2 Identification of Edges (E)

Edges (e_{ij}) are established between pairs of node pixels (vi and vj, where vi and vj belong to V) within the graph. These edges arise when the nodes are connected solely by non-node object pixels. It's feasible for multiple edges to exist between the same pair of nodes, and in such cases, these edges are labeled as eij1, eij2, and so forth. Additionally, a node might possess a self-loop, which is represented as eiL.

To determine the weight assigned to an edge (wij), we take into account the number of diagonal moves (Nd) and the number of horizontal/vertical moves (Nhv) required to traverse from node vi to node vj solely through object pixels. The weight of the edge (wij) is derived from these traversal counts.

$$\omega_{ij} = \omega_d N_d + w_{hv} N_{hv}$$

….eq(2)

3.3 Identification of Strike-out Stroke

We employ a graph-based technique to effectively locate strike-through segments (SS) within the connected component of a struck-out word.

Given a text component image (I), we generate its skeleton (Isk) through the utilization of a thinning algorithm. To eliminate any extraneous branching that may be present, we perform a cleanup process as follows. We identify skeletal pixels with only one neighboring pixel, which we term 'end' or 'terminal' pixels. Additionally, we identify skeletal pixels with three or more neighboring pixels, forming three or more branches; these are referred to as 'junction' pixels. If a direct path exists from a terminal pixel to a junction pixel, and the length of this path is smaller than half of the average stroke width (ST) of the original connected component, we proceed to delete the pixels along this path, including the terminal pixel.

While the thinning algorithm is used for generating the skeleton. Following the pruning of spurious branches as outlined, the resulting shape of the skeleton remains consistent across these effective algorithms. This uniformity arises because handwritten text strokes typically possess a much smaller width in comparison to their length, resulting in highly elongated strokes. Consequently, a well-designed approach for generating the skeleton yields a clean representation that is largely devoid of erroneous branching.

This skeleton structure is represented as an undirected graph, symbolized by an ordered pair G = (V,E), where V signifies the set of nodes, and E represents the set of edges.

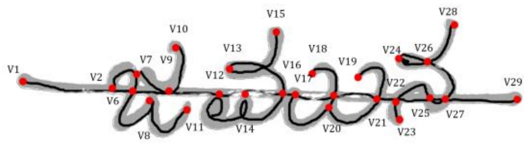


Figure 3: Skeleton image which is divided into 3 regions.

The Figure 3 shows the skeleton image which is divided into 3 regions. The left region, right region and the mid region. The connection between V_1 and V_{28} show the SS.

3.4 Identification of Nodes (V)

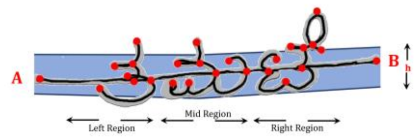
The set of nodes or vertices (V) in the graph G is defined by the terminal pixels and junction pixels present in the skeleton Isk. In situations where nodes are positioned very closely to each other (with a distance less than Tk pixels), we opt for a single node that facilitates left-to-right movement during path estimation. The value of Tk is determined through the ceiling function applied to half of the average stroke width (ST) of the image component I. In a mathematical sense, Tk is calculated as $\lceil ST / 2 \rceil$. It's important to note that Tk adapts to variations in the stroke-image resolution, which in turn influence the average stroke width. As the image resolution increases or decreases, the stroke width also exhibits a corresponding increase or decrease.

3.5 Speed up Approach.

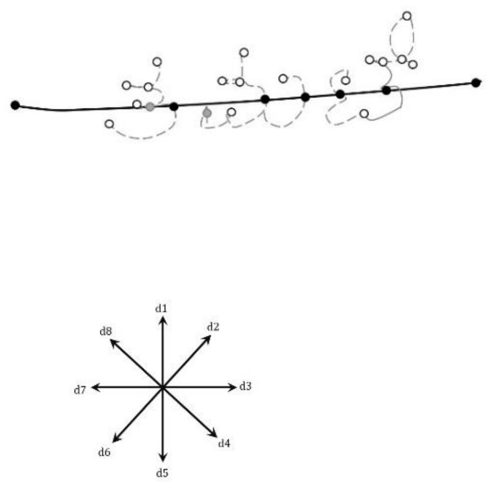
There are 4 steps to analyse the strike outs in speed up approach.

i. The first step is to divide the boundary box into 3 equal components. The 3 equal components are named as a right region, left region and the mid region. In the Figure 4, it is clear that normally the strike out is from left to right. And hence we can avoid the middle region nodes as they do not participate in the shortest path.

ii. If there are any self-loop in the image, they are deleted. The shortest edge is alone considered whenever there are multiple edges between 2 neighbouring notes. And hence the shortest path is calculated.



(a)



(b)

Figure 5: (a) Path from A to B, with the band considered with Height 'h' and Generation of a shortest path using band. (b) 8- neighbour traversal direction.

iii. Considering the shortest path is reasonably street if the 2 nodes are identified between the shortest path are v_i and v_j, and the line v_iv_j joins them both. Then a thick band is created around the line with the thickness of h. The H denotes the tolerance in the deviation from the straightness of shortest path. Hence the head should be selected such a way that it is half the busy text region height.

In the Figure 5 (a) skeleton structure of node edge graph is shown. The black notes and the Grey notes are represents the path from A to B. The white notes are outside the band. Only the black nodes contribute to the final shortest path generation.

iv. In the observation made as there are no retrograde motion whenever there is a striking from left to right direction. So the backtracking of the path is not allowed. This helps in the reduction in number of parts. Hence the movement of the analysis can be done in only One Direction as d1,d2,d3,d4 and D5. There are 6 terminal nodes as, v1,v2,v3,v4,v5 etc.

3.6 Handling multi-word and multi-line strike-out

This is one more case where the user can strike out more than one consecutive word in a single stroke. In this condition, AC is created whose B length is literally greater than the height of the word. In this method, a graph based method which uses directly on such component is used. It is an expensive computation because of many nodes and edges in the graph structure has to be encountered. So the long component is now split into Multiple shorter and small components. Then they are used in graph based scheme in each of the possibility.

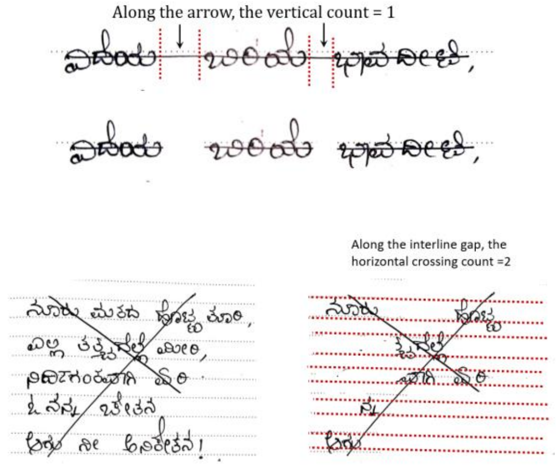


Figure 9: A single strike-out throughout multiple words.

The splitting can be calculated by counting the components that are vertically crossing. This is shown in the Figure 9. A single line or TWO line passes between TWO words. Here the crossing count of ONE or TWO line is detected. And hence consecutive single or double vertical crossing count are found. The region can be split vertically as a component. The Figure 7 shows a long run off single crossing count. The regions between the dashed verticals lines is a pair. The deletion of lines between the dashed line segment throughout the strike out word is also shown in the same Figure 9.

3.7 Recognising a wavy stroke or a zigzag stroke.

One of the difficult and challenging situation in the strikes stroke recognition is when it encounters Azad stroke. The zigzag strokes or the wavy strokes are shown in Figure 6.

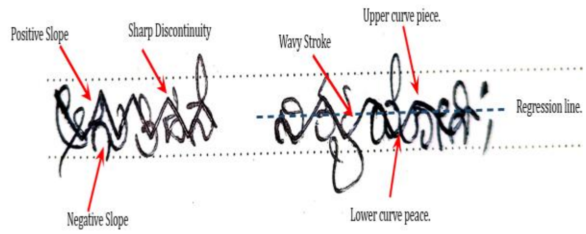


Figure 6: Processing of wavy strike out and zigzag strike out.

The strike out stroke is shown in red colour. In this case we cannot inculcate the shortest path scheme as the path is neither straight nor short. But there are different parts in the image. Different paths can also be obtained using graph based method. Using the algorithms we can verify if the path satisfies the property of zigzag stroke. After reasonable observations made some of the examples, some conclusions can be done. The conclusions are such as strike out is done only on the words and it is not more than 2 characters, These zigzag strokes normally covers the entire character of the world, When the average character height is considered, these exact strokes will lie in the middle of the character height, The characteristics of zigzag stroke is that they have positive slope, negative slope and the sharp slope discontinuity, The last observation is the stroke is normally linear between 2 consecutive slopes and having a discontinuity point at the middle.

To speed up the approach, the number of parts are reduced. The distance between the path from left to right region notes are calculated. The sharp slope and the discontinuity are detected using edge detection technique. The zigzag stroke is confirmed whenever it finds positive slope and a negative slope.

3.8 Handling multiple strike-outs and script-specific strokes

Another unique example of SSs in writing can be seeing in figure 8. The similar procedure to find the strike out word as mentioned in the above can be used here. If the SSs are well spaced, then the detection is very easy. If the SSs are untouched thing, then it is easier to identify the strike out word. At the same time, if there are 3 or more SS drawn on the single word than the identification becomes erroneous. Especially whenever the strokes are very close to each other and they touch each other. In this paper, only 2 strokes are considered.

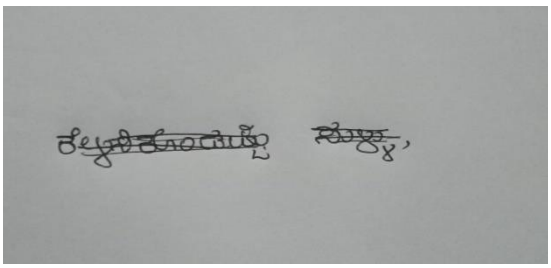


Figure 8: 2 to 3 horizontal SSs on a word.

3.9 Shortest Path Detection.

Let the left region has a terminal node as, V_{L1} and the right region have a terminal node as V_{R1}. It is considered such that all the notes have all The possible pixels as a neighbouring objects. The shortest path is calculated from left to right, using the Dijkstra's algorithm. Using this algorithm few of the shortest paths are identified. Among them the shortest one is taken.

3.10 Recognizing non-horizontal strike-outs

Another example of striking out the words is a slant strike, which is shown in Figure 7. The slant strike may be from left to right or right to left. Based on the observation made, the strike out can be having the following characteristic: It can have a cross mark over the word, It can be a positive slant over the word or a negative slant over the word. If it is a positive slant, it starts from the top right region to bottom left region. If it is a negative slant, it starts from top left region and ends in bottom right region. In the skeletal graph, it is considered only the left to right direction path. So the connected component of the graph.

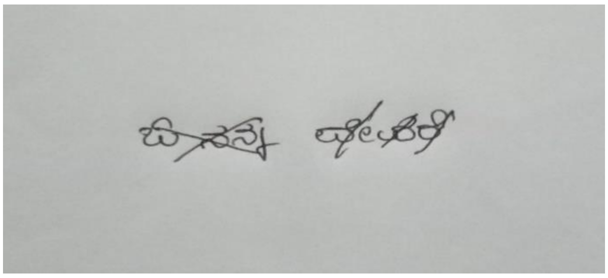


Figure 7: Direction of near vertical and strike outs.

.

In Figure 7, the slope of summer strike outs are nearly vertical. These are used in the detection of single characters. If the strike out is in this manner, then the height of the strike out is larger than the word component. This can be used to find out the SS easily in long text. [28]

3.11 Deletion of Strike-out Stroke

The paper discusses the use of a graph-based approach to detect the skeletal structure of Strike-out Strokes (SS) in a manuscript. By intersecting the text pixels, the approach can identify where the SS cuts through the stroke of characters, allowing for the retention of text pixels and removal of the rest of the SS. If there is overlap with the text tangentially, the thickness of the stroke is measured and uniformity is achieved using an inpainting approach, which fills the gaps in the strikeout structure. The inpainting technique involves selecting a mask and searching for the target region. The paper employs a hybrid sparse representation of deterministic annealing technique for a better extraction of the discrete cosine transform. This method is chosen for the purpose of extraction.

4. RESULTS AND DISCUSSIONS

Experimental result obtained during the conduction of the experiment and the performance of the method for strikeout text detection and their effect is mentioned in this section.

4.1. Strike-out CC detection using SVM

The frequency of strike-out strokes (SS) in the database was examined to gain insights into the various strike-out types present in the writing samples. The table below presents the distribution of these strike-out types among the collected data from the Kannada medium school students. The strike-out types include single, multiple, slanted, crossed, zigzag, wavy, and other forms. Each type reflects the distinct ways in which students corrected or modified their written content. Table 1shows the Frequency of strike-out stroke (SS) types in the database.

The overall precision of 83.33 indicates the accuracy of correctly identifying strike-out words among all detected instances. The overall recall value of 82.61 demonstrates the method's ability to capture the majority of actual strike-out words present in the documents. The calculated overall F-measure of 82.73 presents a balanced assessment, considering both precision and recall, and provides a comprehensive overview of the method's effectiveness in accurately detecting strike-out text in the context of Kannada handwritten documents.

4.2 Preparation of Dataset

The database we utilized for this study consisted of writing samples from around 100 Kannada medium school students, who were roughly 10 years old, with an age range of about ± 3 years. These students were all from Karnataka and were proficient in the Kannada language, having had at least 4 years of writing experience. To gather the samples, we provided the students with information about the study and asked them to write a passage from the famous Kannada poem "Vishvamanava Sandesha" by the well-known poet Sri Kuvempu. The writing was done by hand and included strike-out text. We employed a suitable scanner to convert the handwritten samples into digital format. For further analysis, we processed the collected data using the Python programming language and relevant software libraries.

5. CONCLUSION.

The research focuses on identifying strike-out Kannada text, separating the text, using SVM for classification, and applying inpainting for text recovery. The study examines both trained and untrained databases. Python code with suitable libraries in Anaconda is used for analysis, considering parameters like precision, recall and F1 score. The results show the performance for the existing method with databases, while the proposed methodology shows better results for the standard database, indicating scope for improvement in precision and accuracy.

References.

[1] A.M.Namboodiri and A. K. Jain, “Online script recognition”, IEEE Transactions on Patten Analysis and Intelligence, Vol.26, No.1, pp.124-130, January 2004.

[2] B. Chaudhuri and C. Adak, “An approach for detecting and cleaning of struck-out handwritten text,” Pattern Recognition, vol. 61, pp. 282– 294, 2017.

[3] B. Shi, X. Bai, and C. Yao, “An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition,” IEEE transactions on pattern analysis and machine intelligence, vol. 39, no. 11, pp. 2298–2304, 2016.

[4] Brink, H. van der Klauw, and L. Schomaker, “Automatic removal of crossed-out handwritten text and the effect on writer verification and identification,” in Document Recognition and Retrieval XV, vol. 6815. International Society for Optics and Photonics, 2008, p. 68150A.

[5] C. Adak, B. B. Chaudhuri, and M. Blumenstein, “Impact of struck-out text on writer identification,” in 2017 International Joint Conference on Neural Networks (IJCNN). IEEE, 2017, pp. 1465–1471.

[6] C. Adak and B. B. Chaudhuri, “An approach of strike-through text identification from handwritten documents,” in 2014 14th International Conference on Frontiers in Handwriting Recognition. IEEE, 2014, pp. 643–648.

[7] C. Adak, B. B. Chaudhuri, “Writer Identification from Offline Isolated Bangla Characters and Numerals”, Proc. Int. Conf. on Document Analysis and Recognition (ICDAR), pp.486-490, 2015.

[8] Graves, S. Fernańdez, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in Proceedings of the 23rd international conference on Machine learning. ACM, 2006, pp. 369–376.

[9] In-Jung Kim and Jin Hyung Kim, "Statistical character structure modeling and its application to handwritten Chinese recognition”,

IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 25, No.11, pp.1422-1436, October 2003.

[10] J. C. Aradillas, J. J. Murillo-Fuentes, and P. M. Olmos, “Boosting handwriting text recognition in small databases with transfer learning,” arXiv preprint arXiv:1804.01527, 2018.

[11] J. C. Elizondo-Leal and G. Ramirez-Torres, “An exact euclidean distance transform for universal path planning,” in 2010 IEEE Electronics, Robotics and Automotive Mechanics Conference. IEEE, 2010, pp. 62–67.

[12] J. Puigcerver, “Are multidimensional recurrent layers really necessary for handwritten text recognition?” in 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 1. IEEE, 2017, pp. 67–72.

[13] L. Likforman-Sulem and A. Vinciarelli, “Hmm-based offline recognition of handwritten words crossed out with different kinds of strokes,” 2008.

[14] L. Schomaker, “Advances in Writer Identification and Verification”, Proc. Int. Conf. on Document Analysis and Recognition (ICDAR), vol.2, pp.1268-1273, 2007.

[15] M. Hanmandlu, Pooja Agrawal and Brejesh Lall, "Segmentation of Handwritten Hindi Text: A structural Approach", International Journal of Computer Processing of Languages, Vol. 22, No. 01, pp.167-173, March 2009.

[16] P. Voigtlaender, P. Doetsch, and H. Ney, “Handwriting recognition with large multidimensional long short-term memory recurrent neural networks,” in 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR). IEEE, 2016, pp. 228–233.

[17] R. Manmatha and N. Srimal, “Scale space technique for word segmentation in handwritten documents,” in International conference on scale-space theories in computer vision. Springer, 1999, pp. 22–33.

[18] R. Plamondon, G. Lorette, “Automatic Signature Verification and Writer Identification - The State of the Art”, Pattern Recognition, vol.22, no.2, pp.107-131, 1989.

[19] U. Pal and B. B. Chaudhuri, "Indian Script character recognition: A survey", Pattern Recognition, Vol. 37, pp. 1887- 1899, September 2004.

[20] U.-V. Marti and H. Bunke, “The I am-database: an english sentence database for offline handwriting recognition,” International Journal

on Document Analysis and Recognition, vol. 5, no. 1, pp. 39–46, 2002.

[21] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals,” in Soviet physics doklady, vol. 10, no. 8, 1966, pp. 707–710.

[22] Z. He, X. You, Y.Y. Tang, “Writer Identification of Chinese Handwriting Documents using Hidden Markov Tree Model”, Pattern Recognition, vol.41, no.4, pp.1295-1307, 2008.