Image segmentation of Komering script using bounding box

Muhammad Dio Hamanrora[1], Yesi Novaria Kunang[1], Ilman Zuhri Yadi[1], Mahmud[2]
[1]Intelligent Systems Research Group, Faculty of Science Technology, Universitas Bina Darma, Palembang, Indonesia
[2]Mobile applications developer, Luna Aplikasi Indonesia, Jakarta, Indonesia

Article Info

Article history:

Received Mar 16, 2024
Revised Apr 4, 2024
Accepted Apr 16, 2024

Keywords:

Bounding box
Convolutional neural network
Deep learning
Image segmentation
Komering script

ABSTRACT

The development of deep learning technology is widely used for various purposes, including recognizing characters in a document. One of the scripts that can benefit from this deep learning technology is the Komering script, which is a local script in the South Sumatra region. However, there are challenges in reading documents written in this script, requiring a method to separate each character in a document. Therefore, there is a need for a technology that can automatically segment images of documents written in the Komering script. This research introduces an innovative technique for segmenting images of characters in documents that contain Komering script characters. The segmentation technique employs bounding box technology to separate each Komering script character, subsequently recognized by a pre-trained deep learning model. The bounding box approach imposes restrictions on the segmented object area. To recognize Komering characters, a deep learning model with a convolutional neural network (CNN) algorithm is employed.

Corresponding Author:

Yesi Novaria Kunang
Intelligent Systems Research Group, Faculty of Science Technology, Universitas Bina Darma
Palembang City, South Sumatra, Indonesia
Email: yesinovariakunang@binadarma.ac.id

1. INTRODUCTION

Indonesia, a country with a rich tapestry of regional cultures, boasts remnants of ancient writings, known as scripts, from bygone eras. These scripts exhibit remarkable diversity across the archipelago. For instance, Java is home to the Javanese script (Hanacaraka) [1], Sunda features the Sundanese script [2], Sumatra showcases the Arabic script (Arabic Malay/Jawi) [3], while Lombok possesses the Sasak script [4]. Additionally, Makassar uses the Lontara script [5], Batak employs the Batak script [6], and Komering utilizes the Komering script [7]. These ancient writings are often the focus of scholarly research. Apart from indigenous cultures, foreign scripts like the MODI script [8], Malayalam language script [9], Devanagari script [10], Bangla script [11], Ancient Geez script [12], Chinese [13], Japanese [14], Bengali [15], and Tamil alphabets [16] also captivate the attention of many scholars.

Research on script cultures involves leveraging deep learning technology to preserve these cultures by harnessing modern technological advancements. Deep learning, a subset of machine learning, employs artificial neural networks with multiple layers to identify intricate patterns and execute learning tasks from data, such as facial recognition, automatic translation, and image segmentation [17]. Optical character recognition (OCR) is a prominent example of deep learning technology, enabling computers to identify and translate text from images or scanned documents into editable formats [18]. Several studies, exemplified in script document segmentation [19]–[25], showcase the application of deep learning in preserving script cultures through technological advancements.

Image processing segmenting involves dividing an image into discernible objects or components for further analysis [26]. Various techniques are employed in segmenting script documents, including adaptive thresholding [19], otsu thresholding [20], convolutional neural network (CNN) [21], connected component analysis [22], template matching correlation [23], histogram oriented gradient [24], and bounding box methodologies [25]. The bounding box method defines the image area by delineating objects within the image using small rectangular enclosures around them [27]. To employ the bounding box method for segmentation tasks, a deep learning model, such as a CNN, is essential for object detection and prediction based on the training model utilized [28]. One example of research utilizes CNN models to recognize 336 characters of the Komering script [7]. However, its limitation lies in recognizing each character individually, and it can still be further developed by employing segmentation techniques to identify characters in documents written in the Komering script.

One study employing a line-based text segmentation model for Uyghur-language documents utilized the adaptive thresholding algorithm [19]. The results demonstrated a fairly impressive performance in separating Uyghur text lines with an accuracy of 98.05%. However, this research has limitations, particularly in terms of decreased algorithm performance, especially when dealing with documents containing highly slanted text, making it challenging for the model to separate text accurately. Additionally, the accuracy of this model decreases when there is overlapping text between different text lines in an image. These shortcomings pose a challenge in choosing an appropriate segmentation technique. Further development in research needs to explore alternative approaches that can enhance the model's robustness in addressing the complexities of line segmentation techniques, particularly in the context of Uyghur text.

Another study introduces a novel application of segmentation techniques to identify specific deficiencies such as concrete cracks or handwritten markings on building constructions [25]. The research introduces the development of a faster R-CNN model aimed at detecting defects in buildings, including cracks and handwritten markings on concrete surfaces. Additionally, the study conducts a comparative analysis with the YOLOv2 detection model. While the faster R-CNN model shows promising results in detecting various building defects, the research has limitations in terms of dataset variety. The limitations in dataset variation impact the model's ability to detect objects for identifying diverse building defects.

The conducted research delves into the MODI script, an ancient Indian writing system utilized for Marathi until 1950 [21]. The study developed a model employing a CNN that utilized the CNN autoencoder feature extraction method for character recognition. For classification, a support vector machine (SVM) was incorporated. To enhance data variability and improve generalization, the researchers implemented on-the-fly data augmentation techniques. Impressively, the model attained a remarkable accuracy rate of 99.3%, marking it as one of the most successful attempts in MODI script recognition and showcasing the efficacy of the combined CNN-SVM approach along with data augmentation in this domain.

The advancement of handwritten character recognition (HCR) technology for recognizing handwritten Gujarati text in document images is showcased in this research [22]. The developed models supporting this HCR technology are the CNN and multi-layer perceptron (MLP). The methodology employed for recognizing handwritten Gujarati text involves the use of connected component analysis. Notably, the CNN model achieved the highest accuracy rate of 97.21%, while the MLP model attained 64.48%. However, a critical weakness identified in this study is the presence of inaccuracies in the detected text, signifying the imperative need for further enhancements to improve prediction accuracy.

In a separate investigation, a deep learning model was developed specifically to recognize handwritten Devanagari characters [24]. This study employed histogram of oriented gradients (HOG) features in conjunction with three distinct classification models: SVM, k-nearest neighbor (K-NN), and neural network (NN). Out of these approaches, the SVM + HOG combination yielded the most promising outcome, achieving the highest accuracy rate of 87.38%. The experiment involved a dataset comprising 10,560 images of Devanagari characters, showcasing the potential of the SVM classifier combined with HOG features in accurately identifying handwritten characters within the Devanagari script.

In a recent study, character segmentation of the Gujarati language was investigated employing the otsu thresholding algorithm [20]. The research followed distinct stages encompassing pre-processing, segmentation, character recognition, classification, and evaluation. Notably, the character recognition and feature extraction processes relied on a deep learning neural network (DNN) approach. Encouragingly, the outcomes revealed the system's efficacy in accurately segmenting Gujarati language characters as intended. However, a notable limitation of this study was the inadequate diversity within the dataset utilized, leading to less comprehensive model training.

In a separate study, researchers delved into the development of OCR technology specifically designed for identifying handwritten characters [23]. Their methodology encompassed various approaches, such as the template matching method, statistical techniques, structural pattern recognition, and the introduction of a novel statistical SVM. The primary objective was to create a robust deep learning system capable of accurately recognizing handwritten alphabet letters within images. Notably, the culmination of their efforts yielded promising results, achieving an impressive accuracy rate of 91%.

The research offers novelty in the form of segmentation techniques to separate characters in image documents containing Komering script. The segmentation process itself is crucial for isolating script characters, particularly for the case of the nearly extinct cultural heritage of the Komering script. There are very few individuals who can read this script. Therefore, the use of deep learning technology for recognizing Komering script combined with automatic segmentation techniques using the bounding box method can assist in reading artifacts containing Komering script. This research builds upon previous studies that recognized Komering script using deep learning technology [7]. However, the limitation of previous research lies in recognizing only single-character images and not being able to read image documents containing multiple characters. Therefore, a segmentation technique is needed to separate each character in the document. There have been no researchers utilizing the bounding box segmentation approach to read image documents with Komering script. The bounding box technique used can aid in the separation and recognition of characters, thus significantly contributing to the preservation of cultural heritage.

2. METHOD

This research utilizes the bounding box method to segment characters in image documents containing Komering script. In addition to separating Komering character images, this study also integrates a CNN model trained in previous research [7] as part of its process. This CNN model aids in identifying and predicting recognized Komering characters based on the capabilities acquired through previous training. This combined approach not only delineates the area surrounding the characters but also employs a deep learning model to recognize and predict segmented Komering characters, as depicted in Figure 1. The research process involves data collection, image preprocessing, object segmentation using bounding boxes, and finally recognizing segmented images with the deep learning model.

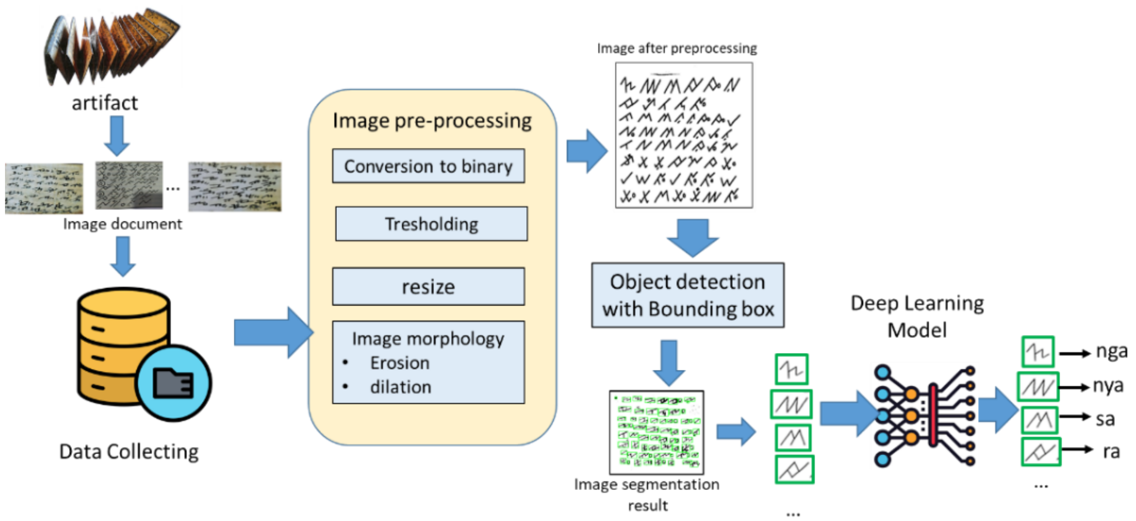


Figure 1. Research flow

2.1. Data collecting

The initial phase involves data collection sourced from the intelligent system research group's google drive folder, housing image files of ancient Komering script letters attributed to the South Sumatra State Museum's collection "Balaputra Dewa." Within this repository, there exist a total of 69 image files. For this research endeavor, a sample dataset was curated, comprising 10 image files specifically from the ancient Komering script letters referenced in the study of "Naskah Kaghas No. INV: 07.47" [29]. Additionally, supplementary datasets were assembled, encompassing 10 re-copies of the Kaghas Manuscript No. INV: 07.47, meticulously transcribed by hand and stored as image files. Thus, the study incorporates a total of 20 Komering script document datasets in the form of image files for analysis and examination.

2.2. Pre-processing

This initial stage encompasses preprocessing an image before segmentation, involving two crucial steps: converting the red, green, and blue (RGB) image to a binary format and implementing thresholding techniques [30]. This process is conducted through image processing methodologies within the Google Colaboratory environment, leveraging the capabilities of the matplotlib and OpenCV libraries. The RGB to binary conversion transforms the image into a binary representation, while thresholding aids in isolating specific elements or features within the image based on predefined criteria. These techniques collectively enhance the image quality and prepare it for subsequent segmentation tasks, ensuring improved accuracy and effectiveness in further analysis or processing steps.

2.2.1. Image to binary conversion

In the process of image analysis, the conversion from an RGB color image to a binary image involves reducing the complexity of the image by representing it with only two colors: black and white [31]. This transformation simplifies the data, as it assigns binary values to pixels-black for one value (often 0) and white for the other (usually 1). The primary objective behind this conversion is to streamline subsequent object identification processes. By limiting the image to binary values, it becomes easier to discern and isolate specific elements or objects within the image, aiding in more efficient and accurate object recognition and analysis.

2.2.2. Thresholding

The described process involves the isolation of objects from their background, streamlining the segmentation procedure by employing a binary threshold [32]. This segmentation technique sets a distinct value, in this case, 127, to differentiate between the object and the background. Pixels with intensity levels below 127 are classified as black, while those above are regarded as white. By applying this binary threshold value, the segmentation process is facilitated, enabling the identification and isolation of the characters comprising the Komering script within the image.

2.2.3. Resize

Resizing is a crucial stage in image processing where images are adjusted to adhere to a standardized size, commonly set at 512×512 pixels [33]. This process aims to ensure uniformity in image dimensions across datasets, thereby enabling a consistent image size for subsequent segmentation procedures. By standardizing the size, resizing facilitates more effective and accurate segmentation processes, allowing for streamlined analysis and comparison of images within the dataset.

2.2.4. Image morphology

Image morphology refers to a vital technique within image processing that alters the shape and structure of objects present in the original image [34]. This study primarily employs two fundamental processes: erosion and dilation. Erosion involves the gradual reduction or thinning of pixels situated at the periphery of objects in a digital image, whereas dilation works in contrast by adding pixels to the boundary of digital image objects, thereby expanding their size [35]. Both erosion and dilation operations utilize a kernel value, a small matrix containing binary values of 1 or 0, also known as structural elements [36]. These kernels are instrumental in adjusting the extent of pixel modifications during the processes, playing a crucial role in the fine-tuning of object attributes within the image.

2.3. Object detection with bounding box

The bounding box technique serves as an approach in segmenting character images by creating a defined enclosure around a specific object within the image using pixel coordinates denoting its upper-right (UR), lower-left (LL), lower-right (LR), and upper-left (UL) corners [37], [38]. This stage involves identifying characters from the Komering script within an image by utilizing object detection techniques. The process begins with segmentation, which entails creating bounding boxes around each recognized Komering script character [39]. More detailed information about the segmentation stage using bounding box can be seen in Figure 2. Detection of these characters is accomplished through the application of a pre-trained CNN model. This model plays a crucial role in assisting the system to detect and highlight identified Komering script characters by outlining them with distinctive green bounding boxes. The CNN model used for segmenting Komering character images is based on a model trained on 336 Komering character samples by Kunang *et al.* [7].

```
1. Define a function read_labels_from_json(json_filename) to read label information
   from a JSON file:
      a. Open the JSON file in read mode.
      b. Load the JSON data.
      c. Close the file.
      d. Return the loaded JSON data.
2. Define a function sort_by_top_left(contour) to sort contours based on their top-
   left coordinates:
      a. Obtain the bounding rectangle of the contour.
      b. Calculate a sorting key based on the top-left coordinates.
      c. Return the sorting key.
3. Load the pre-trained model and label information:
      a. Load the model from the specified path.
      b. Read label information from the specified JSON file.
      c. Convert the label data into a list.
4. Resize the processed image to dimensions (512, 512).
5. Convert the resized image to RGB format for compatibility.
6. Apply thresholding to the resized image to create a binary image using Otsu's
   method.
7. Find contours in the binary image using the external retrieval mode.
8. Sort the contours based on their positions using the sort_by_top_left function.
9. Process each contour in the sorted order:
      a. Obtain the bounding rectangle of the contour.
      b. Check if the width and height of the bounding rectangle are greater than
         10 pixels.
      c. If the width and height satisfy the condition:
            i. Crop the object from the resized image based on the bounding
               rectangle.
           ii. Resize the cropped object to dimensions (48, 48).
          iii. Prepare the object for prediction by expanding its dimensions.
           iv. Predict the label indices using the loaded model.
            v. Retrieve the predicted label and its associated probability.
           vi. Calculate the position for displaying the predicted label text.
          vii. Format the label text with the predicted label and its accuracy
               percentage.
         viii. Draw a green rectangle around the contour on the RGB image.
           ix. ix. Write the label text on the RGB image at the calculated position.
10. Display the RGB image with bounding boxes and predicted labels.
```

Figure 2. Pseudocode for image segmentation using bounding box and predicting characters

2.4. Label prediction and probability accuracy

The identified images undergo a process of label prediction and probability accuracy calculation for recognized Komering script characters. This process involves employing a CNN trained model and a JSON label file to assign labels to the detected characters. Each recognized character is tagged with corresponding accuracy probability percentages, determined by the predictive capabilities of the model utilized. The probability accuracy is computed utilizing the Softmax activation formula, which operates on the output layer of the CNN model. This activation function transforms the output into a probability distribution across potential classes. Mathematically, the formula for calculating the probability with Softmax activation for a specific class is employed to derive the accurate likelihood percentages [40].

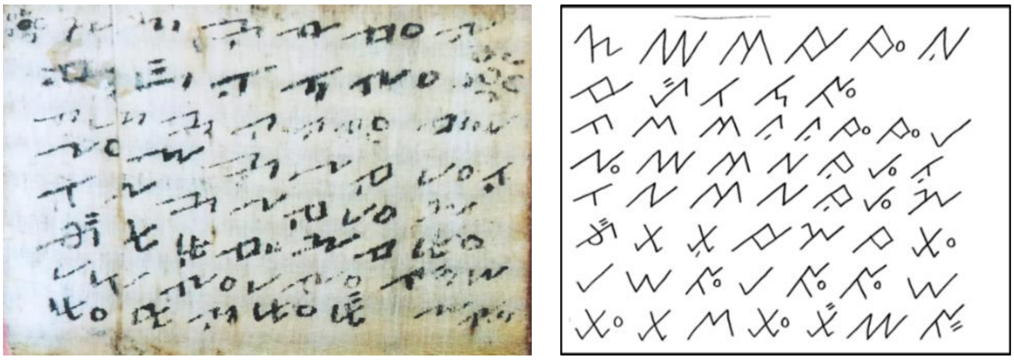
$$P(class = i) = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}} \tag{1}$$

In a classification scenario where an object needs to be assigned to a specific class among multiple possibilities, the probability, denoted as $P(class = i)$, signifies the likelihood that the observed object belongs to a particular class 'i' among the total 'N' possible classes. Each class is associated with an output value, z_i. The probability $P(class = i)$ serves as a measure of confidence or certainty that the object falls within that specific class based on the characteristics or features it exhibits. By evaluating these probabilities across all potential classes, the classification algorithm or model determines the most probable class to which the object belongs, aiding in effective categorization or decision-making processes.

2.5. Result evaluation

The evaluation of segmentation results obtained through the bounding box method is conducted. The accuracy of individual predictions and segmentation outcomes is computed by tallying the total correct predictions against the total number of characters present in each image. This process aims to determine the model's accuracy in image segmentation using bounding boxes.

3. RESULTS AND DISCUSSION
3.1. Dataset
The dataset used in this research consists of 20 image files of Komering script documents sourced directly from the Collection of the South Sumatra State Museum "Balaputra Dewa," specifically Manuscript Kaghas No. INV: 07.47. This dataset is divided into two categories: 10 images displaying the Komering script on ancient wooden backgrounds, showcasing different forms of Komering script characters. Another 10 images feature handwritten Komering script on white paper. The former set showcases the script in its original form, while the latter presents a rewritten version. These documents provide a comprehensive view of the Komering script's variations and styles, offering a diverse dataset for analysis and study purposes. Example dataset images for images taken directly from the manuscript in Figure 3. Figure 3 (a) and images from manuscript copies can be seen in Figure 3(b).



(a) (b)

Figure 3. The example images from the dataset used are as follows (a) direct image from a photo of the ancient artifact and (b) image of a copy of the artifact's inscription

These manuscripts collectively contain a substantial array of Komering script characters. The intended next step involves the detailed processing of these 20 manuscripts to adequately prime them for the subsequent image segmentation phase. Employing a bounding box methodology specifically tailored for the Komering script recognition process.

3.2. Pre-processing
The dataset's images undergo processing to convert them into binary images using thresholding techniques implemented with the OpenCV and Matplotlib libraries in Python on Google Colaboratory. This transformation simplifies the image representation by reducing pixel values to binary (black and white) representations. The primary objective is to streamline subsequent analysis by emphasizing relevant image features. Thresholding, based on color intensity differences and a defined threshold value, effectively converts color images into binary ones, aiding in the identification of edges, shapes, and significant structures within the image. This preprocessing step is crucial for tasks like pattern detection, object segmentation, and character recognition in image processing and computer vision. The original Komering script image and inscription handwritten Komering script image have differing pixel values initially, but after undergoing this preprocessing stage, both types of images will exhibit binary values (black and white).

The pre-processed images exhibit a shift in color to a black and white scheme. Additionally, the pixel values within these images undergo a transformation into binary values, exclusively representing black and white tones. This transformation also extends to the handwritten Komering script inscription images, where initially distinguishable pixel values in black and white become binary values after pre-processing, ensuring uniformity in representing the handwritten content.

In the image processing workflow, the resizing stage involves adjusting the size of the binary image to a standardized dimension of 512×512 pixels using the OpenCV library for resizing and the matplotlib library for displaying the resized image. This standardization simplifies subsequent image processing tasks. The decision to resize images to 512×512 pixels aim to create consistency in handling images with varying sizes across the dataset. Post-resizing, all images from both datasets will exhibit a uniform size of 512×512 pixels, ensuring homogeneity. Both of the original Komering script images and the handwritten inscription ones attain identical sizes subsequent to the resizing process. This suggests the effectiveness of the resizing procedure in standardizing the dimensions across diverse images of Komering script, ensuring uniformity in their final presentation.

3.3. Image morphology in test image

In evaluation process, users are presented with the option to choose their desired image morphology operation, either erosion or dilation. Following the selection of the morphological operation, users can input the desired kernel value or structural element as required. This kernel value becomes the pixel value matrix utilized in the image morphology process to modify the necessary pixel values. Within the dataset used, distinct configurations are available for each image morphology operation conducted. The choice of morphological operation type aligns with the processed image data, allowing for tailored adjustments based on specific requirements.

Erosion was chosen for the original Komering script manuscript due to thick shapes of objects in the dataset. This resulted in less distinct appearances of the Komering script characters in the images, necessitating pixel thinning for clarity. Across the 10 images, a kernel value of 2 was predominantly selected, except for image number 9, which required a kernel value of 4. The decision to opt for kernel value 2 stemmed from extensive experimentation, demonstrating superior prediction outcomes compared to other values for most images. Image number 9, however, due to the notably thick shapes present, demanded a kernel value of 4 to effectively thin out the pixels. The kernel value of 4 was chosen for image number 9 after rigorous testing, which yielded better prediction results compared to other options. The precise details regarding the kernel values assigned to each image of the original Komering script are outlined in the provided Table 1. A kernel value of 2 was used for most images of the Komering script, except for image number 9. The selection of this specific kernel value was the outcome of experimentation involving various kernel values for each image. After thorough testing, the kernel value of 2 stood out as it yielded superior prediction outcomes in comparison to other kernel values.

Table 1. Image morphology operation on binary image of photo the ancient artifact

No	Image morphology type	Kernel
1	Erosion	2
2	Erosion	2
3	Erosion	2
4	Erosion	2
5	Erosion	2
6	Erosion	2
7	Erosion	2
8	Erosion	2
9	Erosion	4
10	Erosion	2

The dilation operation was chosen for the Komering manuscript inscription dataset to address the relatively thin and less distinct shapes of the script characters. It was deemed appropriate given the necessity to thicken pixels for improved distinction. Through multiple experiments with varied kernel values, a kernel value of 2 was eventually selected as it yielded superior prediction outcomes compared to other values. This choice of kernel value facilitated the enhancement of character shapes, contributing to better recognition and analysis of the Komering script within the dataset. The description of kernel values in handwritten Komering script inscription images can be seen in Table 2.

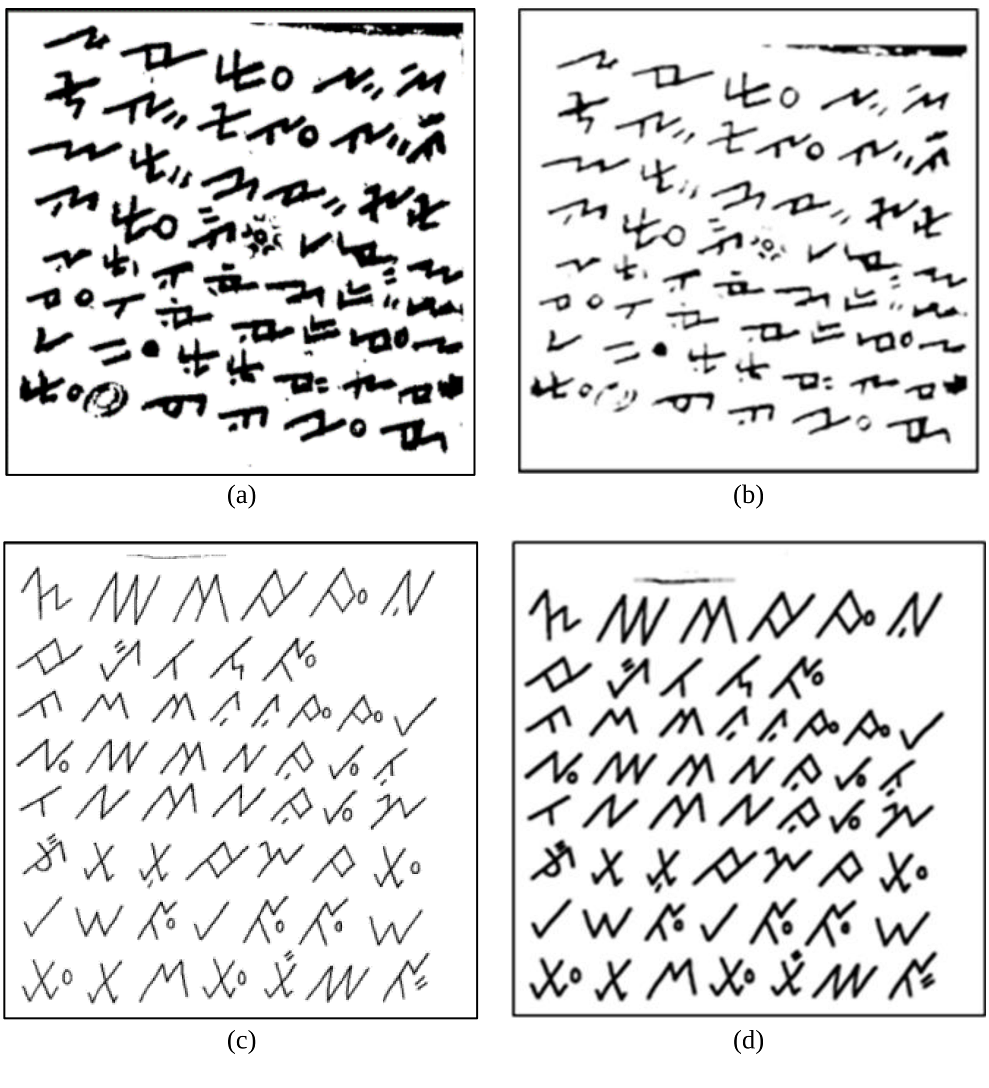
Table 2. Image morphology operation on binary image of artifact’s inscription

No	Image morphology type	Kernel
1	Dilation	2
2	Dilation	2
3	Dilation	2
4	Dilation	2
5	Dilation	2
6	Dilation	2
7	Dilation	2
8	Dilation	2
9	Dilation	2
10	Dilation	2

The dilation operation with a kernel value of 2 was used for all handwritten Komering script inscription images, chosen after thorough experimentation to improve predictive accuracy. After conducting numerous experiments, it became apparent that a kernel value of 2 consistently generated superior predictions compared to other values. This choice was pivotal in enhancing the overall quality of the predictions made from the handwritten Komering script inscription images.

In both the original Komering script and the handwritten Komering script inscription, distinct alterations are observed following image morphology operations. In the original Komering script, characters undergo pixel thinning subsequent to the erosion operation, refining the details and reducing the thickness of the character objects. Conversely, in handwritten Komering script inscription images subjected to dilation operations, there’s an observable pixel thickening surrounding the objects representing Komering script characters. The visual transformation before and after the erosion and dilation processes can be examined in Figure 4.

In Figures 4(a) and (b), a noticeable transformation is evident before and after the erosion operation on the image. Initially, in Figure 4(a), the Komering script characters appear thicker. However, following the erosion process using a specific kernel value, the resulting image in Figure 4(b) demonstrates pixel thinning, rendering the characters clearer and more defined. Conversely, Figures 4(c) and (d) illustrate the changes pre- and post-dilation. Initially, in Figure 4(c), the Komering script characters exhibit relatively thin pixel values, presenting slender appearances. Subsequent dilation, depicted in Figure 4(d), results in pixel thickening, enhancing the clarity and prominence of the characters. These operations, erosion and dilation, respectively refine and enhance the visual representation of the Komering script characters in the images.



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

Figure 4. Morphological image processing results of; (a) before erosion, (b) after erosion, (c) before dilation, and (d) after dilation

3.4. Object detection with bounding box

In Object detection phase, the Komering script characters depicted in the image are detected and assigned bounding boxes using a deep learning CNN model specifically trained for recognizing these characters. The object detection process employs the trained model to identify and localize the Komering script letters within the image, utilizing OpenCV for object detection and bounding box creation. Subsequently, the matplotlib library is utilized to visualize the detected objects within the image. Following this, the stage involves the segmentation of Komering script characters using the bounding box method. Utilizing the previously trained CNN model, the segmentation process involves delineating square-shaped boundaries around each successfully detected Komering script character within the image. These boundaries, showcased in green, encircle each identified object recognized as a Komering script character. The images both before and after the segmentation process showcasing the delineated Komering script characters can be observed in Figure 5. These bounding lines are rendered in green and exclusively enclose the objects accurately identified as Komering script characters.

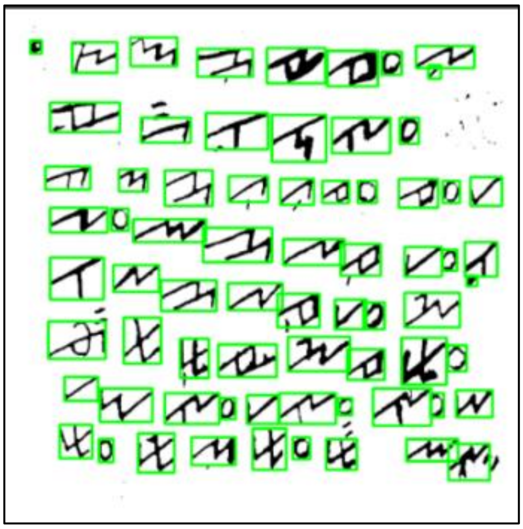


Figure 5. The results of object detection in image Komering script after object detection

3.5. Result of bounding box segmentation

A CNN training model is utilized to accurately recognize objects as Komering script characters. Subsequently, a JSON label file, containing Latin letter labels corresponding to the Komering script characters trained in the CNN model, is employed for predictive label information. Each detected object recognized as a Komering script character is annotated with a Latin letter label and an accuracy probability description based on the CNN model’s performance and the JSON label file used. The JSON file encompasses 336 Latin letter labels for the Komering script characters. Post-process, these recognized objects are highlighted with green bounding boxes, with accompanying blue-colored labels displaying the Latin letter identified and its corresponding probability accuracy.

In this phase, a sequential display of cropped images derived from object detection, along with labeled predictions and probability accuracy, is executed through visualization. The process involves showcasing image segmentation outcomes specifically for Komering script characters. Using the matplotlib library, the segmented images are exhibited alongside descriptions indicating the predicted label and corresponding probability accuracy. Figure 6 illustrates the segmentation results for handwritten Komering script.

Notably, the probability accuracy in segmenting the handwritten Komering script inscription surpasses that of the original dataset. This discrepancy arises due to the CNN model’s enhanced proficiency in character recognition within the handwritten inscription dataset as opposed to the original Komering manuscript dataset. The CNN’s superior ability to discern and interpret the nuances of characters in the handwritten dataset leads to the observed higher probability accuracy, underscoring its efficacy in handling handwritten variations of the Komering script.

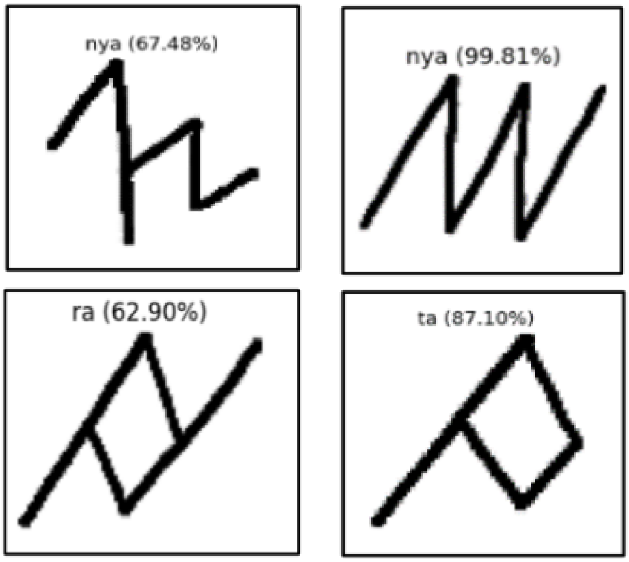


Figure 6. The result of segmentation on handwritten Komering script image

3.6. Result evaluation

At this stage, an assessment is underway to evaluate the segmentation outcomes of the Komering script characters utilizing the bounding box technique. The segmentation accuracy for both datasets is being calculated manually by determining the count of accurately predicted Komering script characters against the total number of characters within each image. This process aligns with the methodologies outlined in referenced sources [41], [42]. In the initial dataset comprising original Komering script manuscripts, the overall accuracy varies significantly, resulting in relatively low average accuracy. The segmentation truth values for the original Komering script manuscript dataset is detailed in Table 3.

Table 3. Image morphology operation on binary image of Komering manuscript

Image number	Total number of characters	Total number of bounding boxes	Total correct bounding boxes	Correct predictions	Percentage
1	54	105	54	33	61.11%
2	48	127	47	28	58.33%
3	40	100	40	25	62.5%
4	38	110	38	21	55.26%
5	29	122	29	8	27.58%
6	35	132	36	12	34.28%
7	39	107	39	9	23.07%
8	36	87	36	5	13.88%
9	49	123	46	17	34.69%
10	52	146	51	2	3.84%

With an overall average correct prediction of 37.45%. Based on the findings presented in the tables, the average segmentation accuracy for the initial dataset appears to be notably low. Among the ten images in this dataset, the highest accuracy, at 62.5%, is observed in Table 3. This suggests that the model's performance in predicting Komering script characters within this dataset is somewhat unsatisfactory. The low prediction results on images of original artifacts are mainly due to the fact that photos of original artifacts, which are typically made of wood, bamboo blades, or buffalo horns, often have dark and shadowy backgrounds. Consequently, other objects may be detected. Additionally, for writing on artifacts made of wood, many characters are blurry and unclear, making detection difficult.

However, transitioning to the second dataset containing handwritten Komering script manuscripts inscription, there is a contrasting trend with relatively higher truth value accuracy. Across the ten images in this second dataset, varying truth value accuracy figures are recorded, indicating a more favorable performance compared to the first dataset. Detailed information regarding the truth value accuracy for the second dataset is provided in the accompanying Table 4.

Table 4. Image morphology operation on binary image of Komering manuscript inscription

Image number	Total number of characters	Total number of bounding boxes	Total correct bounding boxes	Correct predictions	Percentage
1	54	84	54	46	85.18%
2	48	76	47	39	81.25%
3	40	68	40	34	85%
4	38	86	38	29	76.31%
5	29	56	29	21	72.41%
6	35	55	36	29	82.85%
7	39	67	39	26	66.66%
8	36	66	36	26	72.22%
9	49	76	48	33	67.34%
10	52	92	52	32	61.53%

Based on the table's data, the recorded accuracy values exhibit notably high percentages. The highest accuracy is notably seen in image 1, achieving an 85.18% prediction rate in correctly identifying characters. The model demonstrates commendable proficiency in recognizing Komering script characters within the second dataset, showcasing consistently high accuracy levels across all images. These figures collectively suggest a strong capability in accurately identifying and deciphering Komering script characters within the dataset, indicating the model's effectiveness in this specific task. The overall average prediction result for this second dataset is 75.08%. The results of detecting Komering characters using bounding box segmentation techniques yield fairly good prediction results. However, due to the lack of prior research on reading documents written in Komering script or Ulu script, the obtained results cannot be fully utilized. Nevertheless, some studies on character recognition from documents, such as in the case of MODI script using the vertical projection profile technique achieving only 67% [8], and Lontar character recognition with segmentation also employing bounding box techniques reaching an accuracy of 75% [5]. Our research has yielded promising results. However, further exploration is still needed to address the limitations of the bounding box segmentation method, especially regarding box overlap issues and the abundance of diacritics on Komering characters. Other character segmentation approaches for different script types that yield better results should be considered for future research. Approaches such as separating diacritic characters in Bangla script, as done in previous research [11], or employing OCR techniques [10], [18], could be applied to enhance character recognition in documents.

The findings of this study partly support the initial hypothesis, especially as the overall prediction results obtained are quite in line with expectations. However, these findings also indicate various shortcomings and weaknesses that need to be considered. One key factor contributing to suboptimal findings is the chosen bounding box segmentation method and CNN model, which still have some inherent limitations. One such limitation is the methods' tendency to interpret closely situated or merged objects as a single entity during detection, which can result in inaccurate object delineation. Additionally, the variation in the shapes of Ulu script characters in each region also contributes to prediction inaccuracies. For example, in testing several characters found on the studied artifacts, they are slightly different from the characters in the dataset used in the pretrained model from previous research [7]. Therefore, further detailed research is needed to develop a model that can recognize various variations of Ulu script to address this limitation. Although these findings largely meet the expected outcomes, it is important to remember that there are still some shortcomings and weaknesses that need to be addressed. Hence, this study underscores the importance of continued exploration and refinement in the development of Komering script character recognition techniques to achieve more optimal results in the future.

It is important to consider the consequences of avoiding the problem being faced in recognizing Komering script characters. Ignoring this issue can result in difficulty in obtaining information from documents written in this script, and it may hinder efforts to preserve and understand the Komering cultural heritage. Moreover, as the ability to read and understand historical documents diminishes, there may be a decrease in understanding of Komering culture and history as a whole. Therefore, addressing this issue effectively is crucial to maintaining the relevance of Komering cultural and ensuring accessibility for future generations.

4. CONCLUSION

Our study highlights the challenges and potential solutions in segmenting and recognizing Komering script characters using bounding box segmentation techniques alongside CNN models. In our research, we evaluated the bounding box method for the segmentation process of Komering script images. The segmentation results were assessed using CNN models to recognize characters in documents across two different datasets. The accuracy results varied significantly between the two datasets. In the dataset of images of ancient manuscript artifacts on wooden bark, the results were not satisfactory due to the unclear quality of the writing on the artifacts. However, when applied to a dataset of handwritten copies of Komering script artifacts, the accuracy values showed fairly good performance with an average accuracy of 75.08%. The main challenge of segmenting Komering characters using bounding box techniques is the overlapping segmentation results and occasionally separating diacritical marks. To address the challenge of detecting irrelevant objects, future research will develop dynamic bounding box models that allow for adaptive adjustments to obtain desired results. Additionally, several advanced models such as YOLO, single shot detector, or fast mask R-CNN could be evaluated to overcome the challenge of detecting irrelevant objects. While our results show promising accuracy rates, especially in handwritten Komering script images, there remains room for improvement. Future research should focus on exploring advanced models and refining segmentation methods to enhance accuracy and address challenges such as detecting irrelevant objects. The researchers hope that by refining the use of image segmentation technology combined with deep learning techniques, as demonstrated, we can ensure better preservation and accessibility of cultural heritage, particularly the Komering script, for future generations.

ACKNOWLEDGEMENTS

Many thanks to my research colleagues who have assisted throughout this research endeavor. We would also like to express gratitude to the intelligent systems research group Universitas Bina Darma for their support in facilitating this research.

REFERENCES

[1] A. Setiawan, A. S. Prabowo, and E. Y. Puspaningrum, "Handwriting character recognition javanese letters based on artificial neural network text mining and text information retrieval view project artificial intelligence view project handwriting character recognition javanese letters based on artificial neural network," *Network Security and Information System (IJCONSIST)*, vol. 1, no. 1, pp. 39–42, 2019.

[2] I. Maliki and A. Febriansyah, "Implementation of convolutional neural network - extreme learning machine for handwriting recognition of sundanese script," *ICSPIS 2023 - Proceedings of the 9th International Conference on Signal Processing and Intelligent Systems*, vol. 18, no. 2, pp. 1113–1123, 2023, doi: 10.1109/ICSPIS59665.2023.10402761.

[3] F. Arnia, K. Saddami, and K. Munadi, "Moment invariant-based features for Jawi character recognition," *International Journal of Electrical and Computer Engineering*, vol. 9, no. 3, pp. 1711–1719, 2019, doi: 10.11591/ijece.v9i3.pp1711-1719.

[4] M. Tajuddin *et al.*, "Baluk olas (eighteen) sasak scripts in the digital era based on the mobile games," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 13, no. 3, pp. 1000–1017, 2023, doi: 10.18517/ijaseit.13.3.17019.

[5] A. Hidayat, I. Nurtanio, and Z. Tahir, "Segmentation and recognition of handwritten Lontara characters using convolutional neural network," in *2019 International Conference on Information and Communications Technology, ICOIACT 2019*, 2019, pp. 157–161, doi: 10.1109/ICOIACT46704.2019.8938445.

[6] F. Tambunan, E. Ginting, E. V. Haryanto, and M. Fauzi, "Pattern recognition of batak script using habbian method," *2020 8th International Conference on Cyber and IT Service Management, CITSM 2020*, pp. 18–21, 2020, doi: 10.1109/CITSM50537.2020.9268839.

[7] Y. N. Kunang, I. Z. Yadi, Mahmud, and M. Husin, "A new deep learning-based mobile application for komering character recognition," in *2022 5th International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2022*, May 2022, pp. 294–299, doi: 10.1109/ISRITI56927.2022.10053072.

[8] P. A. Tamhankar, K. D. Masalkar, and S. R. Kolhe, "A novel approach for character segmentation of offline handwritten marathi documents written in MODI script," *Procedia Computer Science*, vol. 171, no. 2019, pp. 179–187, 2020, doi: 10.1016/j.procs.2020.04.019.

[9] K. Manjusha, M. A. Kumar, and K. P. Soman, "On developing handwritten character image database for Malayalam language script," *Engineering Science and Technology, an International Journal*, vol. 22, no. 2, pp. 637–645, 2019, doi: 10.1016/j.jestch.2018.10.011.

[10] B. Dessai and A. Patil, "A deep learning approach for optical character recognition of Handwritten Devanagari script," *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies, ICICICT 2019*, pp. 1160–1165, 2019, doi: 10.1109/ICICICT46008.2019.8993342.

[11] N. Majid and E. H. B. Smith, "Segmentation-free bangla offline handwriting recognition using sequential detection of characters and diacritics with a faster R-CNN," *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, pp. 228–233, 2019, doi: 10.1109/ICDAR.2019.00045.

[12] F. A. Demilew and B. Sekeroglu, "Ancient Geez script recognition using deep learning," *SN Applied Sciences*, vol. 1, no. 11, pp. 1–7, 2019, doi: 10.1007/s42452-019-1340-4.

[13] Z. Cao, J. Lu, S. Cui, and C. Zhang, "Zero-shot Handwritten Chinese character recognition with hierarchical decomposition embedding," *Pattern Recognition*, vol. 107, p. 107488, 2020, doi: 10.1016/j.patcog.2020.107488.

[14] S. Saini and V. Verma, "Japanese historical character recognition using deep convolutional neural network (DCNN) with dropblock regularization," *International Journal of Recent Technology and Engineering*, vol. 8, no. 2, pp. 3510–3515, 2019, doi: 10.35940/ijrte.B2923.078219.

[15] M. S. Islam, M. M. Rahman, M. H. Rahman, M. W. Rivolta, and M. Aktaruzzaman, "RATNet: a deep learning model for Bengali handwritten characters recognition," *Multimedia Tools and Applications*, vol. 81, no. 8, 2022, doi: 10.1007/s11042-022-12070-4.

[16] M. A. Pragathi, K. Priyadarshini, S. Saveetha, A. S. Banu, and K. O. M. Aarif, "Handwritten Tamil character recognition using deep learning," *Proceedings - International Conference on Vision Towards Emerging Trends in Communication and Networking, ViTECoN 2019*, pp. 1–5, 2019, doi: 10.1109/ViTECoN.2019.8899614.

[17] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," *Electronic Markets*, vol. 31, no. 3, pp. 685–695, 2021, doi: 10.1007/s12525-021-00475-2.

[18] J. Memon, M. Sami, R. A. Khan, and M. Uddin, "Handwritten optical character recognition (OCR): a comprehensive systematic literature review (SLR)," *IEEE Access*, vol. 8, pp. 142642–142668, 2020, doi: 10.1109/ACCESS.2020.3012542.

[19] E. Suleyman, A. Hamdulla, P. Tuerxun, and K. Moydin, "An adaptive threshold algorithm for offline Uyghur handwritten text line segmentation," *Wireless Networks*, vol. 27, no. 5, pp. 3483–3495, 2021, doi: 10.1007/s11276-019-02221-1.

[20] S. Aniket, R. Atharva, C. Prabha, D. Rupali, and P. Shubham, "Handwritten Gujarati script recognition with image processing and deep learning," *2019 International Conference on Nascent Technologies in Engineering, ICNTE 2019 - Proceedings*, no. Icnte, pp. 1–4, 2019, doi: 10.1109/ICNTE44896.2019.8946074.

[21] S. Joseph and J. George, "Handwritten character recognition of MODI script using convolutional neural network based feature extraction method and support vector machine classifier," *2020 IEEE 5th International Conference on Signal and Image Processing, ICSIP 2020*, pp. 32–36, 2020, doi: 10.1109/ICSIP49896.2020.9339435.

[22] J. Pareek, D. Singhania, R. R. Kumari, and S. Purohit, "Gujarati Handwritten character recognition from text images," *Procedia Computer Science*, vol. 171, no. 2019, pp. 514–523, 2020, doi: 10.1016/j.procs.2020.04.055.

[23] Y. B. Hamdan and Sathish, "Construction of statistical SVM based recognition model for Handwritten character recognition," *Journal of Information Technology and Digital World*, vol. 3, no. 2, pp. 92–107, 2021, doi: 10.36548/jitdw.2021.2.003.

[24] S. P. Deore and A. Pravin, "Histogram of oriented gradients based off-line handwritten devanagari characters recognition using SVM, K-NN, and NN classifiers," *Revue d'Intelligence Artificielle*, vol. 33, no. 6, pp. 441–446, 2019, doi: 10.18280/ria.330606.

[25] J. Deng, Y. Lu, and V. C. S. Lee, "Concrete crack detection with handwriting script interferences using faster region-based convolutional neural network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 35, no. 4, 2020, doi: 10.1111/mice.12497.

[26] X. Liu, L. Song, S. Liu, and Y. Zhang, "A review of deep-learning-based medical image segmentation methods," *Sustainability (Switzerland)*, vol. 13, no. 3, pp. 1–29, 2021, doi: 10.3390/su13031224.

[27] D. T. Delight and V. Karunakaran, "Deep learning based object detection using mask RCNN," *Proceedings of the 6th International Conference on Communication and Electronics Systems, ICCES 2021*, no. August 2021, pp. 1684–1690, 2021, doi: 10.1109/ICCES51350.2021.9489152.

[28] S. Wani, D. Upadhyay, A. Warise, S. Ladhe, and S. Sanas, "Implementation of deep learning techniques for image segmentation and object recognition," *SSRN Electronic Journal*, 2022, doi: 10.2139/ssrn.4111884.

[29] M. A. Ridhollah, N. U. Kalsum, and S. Khudin, "Ulu Manuscript: Traditional Medicines in Kaghas Manuscript Number. Inv 07. 47 South Sumatra State Museum Collection (Philological Studies) *in Indonesian*," *Tanjak: Sejarah dan Peradaban Islam*, vol. 1, no. 3, pp. 70–90, 2021, doi: 10.19109/tanjak.v1i3.9704.

[30] V. K. Tewari, C. M. Pareek, G. Lal, L. K. Dhruw, and N. Singh, "Image processing based real-time variable-rate chemical spraying system for disease control in paddy crop," *Artificial Intelligence in Agriculture*, vol. 4, pp. 21–30, 2020, doi: 10.1016/j.aiia.2020.01.002.

[31] G. Ramesh, J. Logeshwaran, and K. Rajkumar, "The smart construction for image pre-processing of mobile robotic systems using neuro fuzzy logical system approach," *NeuroQuantology*, vol. 20, no. 10, pp. 6354–6367, 2022, doi: 10.14704/nq.2022.20.10.NQ555629.

[32] S. Jardim, J. António, and C. Mora, "Image thresholding approaches for medical image segmentation-short literature review," *Procedia Computer Science*, vol. 219, pp. 1485–1492, 2023, doi: 10.1016/j.procs.2023.01.439.

[33] S. H. Nam *et al.*, "Content-aware image resizing detection using deep neural network," *Proceedings - International Conference on Image Processing, ICIP*, vol. 2019-September, pp. 106–110, 2019, doi: 10.1109/ICIP.2019.8802946.

[34] Y. Hou *et al.*, "The state-of-the-art review on applications of intrusive sensing, image processing techniques, and machine learning methods in pavement monitoring and analysis," *Engineering*, vol. 7, no. 6, pp. 845–856, 2021, doi: 10.1016/j.eng.2020.07.030.

[35] S. K. Sharma and B. Chourasia, "Review paper on segmentation of color image using morphological processing," *International Journal of Computer Sciences and Engineering*, vol. 7, no. 7, pp. 276–279, 2019, doi: 10.26438/ijcse/v7i7.276279.

[36] G. Boato, D. T. Dang-Nguyen, and F. G. B. De Natale, "Morphological filter detector for image forensics applications," *IEEE Access*, vol. 8, pp. 13549–13560, 2020, doi: 10.1109/ACCESS.2020.2965745.

[37] S. Bonechi, P. Andreini, M. Bianchini, and F. Scarselli, *Generating Bounding Box Supervision for Semantic Segmentation with Deep Learning*, vol. 11081 LNAI. Springer International Publishing, 2018.

[38] N. R. Nusantika, X. Hu, and J. Xiao, "Newly designed identification scheme for monitoring ice thickness on power transmission lines," *Applied Sciences (Switzerland)*, vol. 13, no. 17, 2023, doi: 10.3390/app13179862.

[39] L. Jiao *et al.*, "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, pp. 128837–128868, 2019, doi: 10.1109/ACCESS.2019.2939201.

[40] Z. Ul Abideen *et al.*, "Uncertainty assisted robust tuberculosis identification with bayesian convolutional neural networks," *IEEE Access*, vol. 8, pp. 22812–22825, 2020, doi: 10.1109/ACCESS.2020.2970023.

[41] R. Nuzulur, *Guidelines for Writing Ulu Script, South Sumatra in Indonesian*, Yogyakarta: KBM Indonesia, 2021.

[42] M. M. Syarifuddin, "Towards an Encoding for Surat Ulu," pp. 1–11, 2021.

BIOGRAPHIES OF AUTHORS

Muhammad Dio Hamanrora, S.Kom. is an alumnus of the Information Systems program at Bina Darma University. He hails from Tulung Selapan Village in Ogan Komering Ilir Regency, South Sumatra Province. Pursuing his degree in Information Systems at the Faculty of Science and Technology, he extensively engages in machine learning, programming, design, and English language studies within his curriculum. Dio actively participates in campus organizations with the aim of enhancing his interpersonal skills and gaining practical experience. He can be contacted at email: m.diohamanrora@gmail.com.

Dr. Yesi Novaria Kunang, S.T., M.Kom. obtained her Bachelor's degree (ST) in Electrical Engineering from Sriwijaya University. She then pursued a master's degree in Computer Science at Gadjah Mada University, earning the title (M.Kom). She completed her doctoral program in the field of Engineering, specializing in Computer Science at the University. She has been a lecturer in the Information Systems Program at Bina Darma University since 2000 until now. She has served as a supervisor and co-supervisor at the master's level and as a co-supervisor for several Ph.D. students. Currently, she is the chair of the Intelligent Systems Research Group at Bina Darma University, focusing on research in Intelligent Systems, deep learning, machine learning, and Information Security. She has produced more than 170 research articles in the form of proceedings and national and international journal articles. Additionally, she actively serves as a reviewer for various national and international journal articles. She can be contacted at email: yesinovariakunang@binadarma.ac.id.

Ilman Zuhri Yadi, M.Kom., M.M. is a lecturer in the Information Systems program at Bina Darma University. He completed his undergraduate studies in Information Systems at Bina Darma University in 1999. He earned his Master's degree in Management at Bina Darma University, and later pursued another Master's degree in Computer Science at Bina Darma University as well. Currently, he is an active researcher in the Intelligent Systems Research Group at Bina Darma University. His publications in the field of computer science focus on software development, information security, and data science. He can be contacted at email: ilmanzuhriyadi@binadarma.ac.id.

Mahmud, S.Kom. is an alumnus of the Information Systems program at Bina Darma University. He specializes in full-stack development with primary expertise in mobile development. He excels in building robust and user-friendly applications for both Android and iOS platforms. Additionally, he has a passion for delivering high-quality solutions and a dedication to staying updated with the latest technologies. Currently, he is actively involved as a developer in comprehensive mobile and web development. As a researcher, he has contributed to the creation of several copyrighted products and publication articles with the Intelligent Systems Research Group at Bina Darma University. His research focuses on intelligent systems and mobile development. He can be contacted at email: mahmud120398@gmail.com.