Classification of Bugis and Makassar Lontara Texts with Viterbi Algorithm

Andi Hutami Endang1, Achmad Zulfajri Syaharuddin2

1hutamiendang@kallainstitute.ac.id, 2ajisan@kallainstitute.ac.id
1,2Insitut Teknologi dan Bisnis Kalla

Abstract
The Bugis and Makassar ethnic groups are among those originating from South Sulawesi, characterized by the use of the Lontara Bugis and Makassar scripts. The data utilized in this research consists of image data as the initial input. Subsequently, preprocessing is conducted where the data is cropped to obtain 168x168 pixel dimensions for each word. The cropped images are then labeled. In this study, labeling involves the conversion of Lontara script into Latin words. The same procedure is applied to each letter. Next, the obtained words are categorized into two classes: Bugis language and Makassar language. The next stage is processing, where the Viterbi algorithm is employed to determine the priority within a sentence used in the training and testing processes. Python programming language is used in this research. The research utilizes image data processed in the aforementioned preprocessing stage. The data transformed from Lontara script into Latin words is used in the Viterbi process. The Viterbi algorithm tests the data from the inputted sentences. Based on the research results, it is concluded that the use of the Viterbi algorithm in classifying Bugis Makassar Lontara texts yields accurate probability results. However, there are some aspects to be considered. The more data utilized, the better the classification results will be.

Keywords: Classification, Lontara, Viterbi Algorithm

This is an open access article under the CC BY-SA license

Abstrak

Kata kunci: Klasifikasi, Lontara, Algoritma Viterbi

This is an open access article under the CC BY-SA license
1. Introduction

The Bugis and Makassar ethnic groups, originating from South Sulawesi, are distinguished by their unique script, commonly referred to as the Lontara Bugis and Makassar script. The utilization of the Lontara script to document the histories of local kingdoms in South Sulawesi demonstrates the historical significance of using regional languages, specifically the Bugis language. Initiatives in learning and literacy that focus on historical texts represent fundamental steps towards rekindling the appreciation of local culture among educated individuals.

The Lontara script features unique edge patterns. Identifying these patterns requires focusing attention on a specific pattern for recognition [1]. Local classification is effectively utilized to handle variations in luminance within images. The primary reason for this is the failure to recognize specific characters, texts with certain properties, or overlays of particular elements that occur, as these are specific to technical images [2].

Text is recognized using directional features extracted from the character regions using a non-linear mesh [3]. Understanding text from images encompasses two primary tasks: text detection and text recognition [4]. The detection and recognition of handwritten text in natural images are more complex and challenging due to variations in writing styles, placing text over other text, and other complexities [5].

Numerous approaches have been employed to achieve significant results using machine learning algorithms [6]. Consequently, researchers have attempted a previously unexplored method by utilizing the Viterbi algorithm. The aim of this research is to classify texts in the Bugis and Makassar regional languages. Through the classification of these regional languages, the general public can discern the differences between them.

2. Method

The data used in this study consist of image data as the initial input. Subsequent preprocessing involves cropping the data to obtain a size of 168x168 pixels for each word. The cropped images are then labeled. The labeling in this study involves converting Lontara script into Latin script, with each character undergoing the same process. The languages obtained are then grouped into two categories: Bugis and Makassar. The next stage is processing, where the Viterbi algorithm is used to determine priorities within a sentence used in the training and testing processes. Here are the stages of the regional language classification research:

![Stages of research Regional Language Classification](Picture 1. Stages of research Regional Language Classification)

2.1. Early Research

The initial phase of the research includes a literature study to identify and define the research object and to analyze potential problems that warrant investigation. Based on the outcomes of the preliminary research, the identified issue is the difficulty that the general public faces in understanding the Lontara text of the Bugis and Makassar regions. At this stage, researchers also conduct a literature review, seeking references and reviewing previous studies.

The research conducted by Dea Delia et al., 2015 [7], designed an application that facilitates the understanding of the Sundanese language from West Java. This study identified Sundanese script using edge detection and the LVQ (Learning Vector Quantization) method. The research achieved an
accuracy of 60.90% for 10 syllables without cropping. The highest accuracy, 61.53%, was obtained for 10 words with cropping, while the lowest accuracy, with 30 syllables, was 17.78%.

The research conducted by I Gede Susrama Mas Diyasa and Romadhon in 2023 [8] developed a program with capabilities in image processing and CNN (Convolutional Neural Network) machinery for character recognition and classification of Javanese script. The software developed utilizes contour detection and smart edge detection using the Java OpenCV library for character image segmentation. The CNN then classifies the segmented Javanese script into 20 classes. For evaluation, four segmentation models were created for classification: the canny model, contour model, canny+contour model, and contour+canny model. These models were compared in terms of accuracy, loss, and runtime. From this study, the highest accuracy was achieved by the contour+canny segmentation model, with an accuracy of 0.5307 and a loss of 5.5167. Conversely, the lowest accuracy was observed in the canny+contour model, with an accuracy of 0.4135 and a loss of 5.1892.

The study conducted by Eka et al. in 2014 [9] employed the Hidden Markov Model (HMM) to detect handwritten text. This research utilized the Modified Direction Feature (MDF), which is a combination of the Direction Feature (DF) and Transition Feature (TF). MDF involves capturing and calculating character trait values based on the strokes from various directions, making the character features unique. Once these character features are identified, they are classified using the Hidden Markov Model, which is an extension of the Markov chain. In HMM, the states are not directly observable (hidden); instead, one can only observe the variables that are influenced by these states. The research by Eka et al. in 2014 achieved an average best accuracy of approximately 74.72% for character recognition, with an average system testing computational time of 2.23 seconds.

In another study conducted by Daming Shi, Steve R. Gunn, and Robert I. Dumper in 2002 [1], the Viterbi Algorithm was utilized for recognizing characteristics of handwritten Chinese characters. This research arranged the composition of characters by finding the optimal path using the Viterbi Algorithm. The algorithm processes to find the best tag value for a word by calculating the maximum value from the transition calculations combined with emission probabilities derived from Hidden Markov Model modeling [10].

Similarly, Himani Kohli, Jyoti Agarwal, and Manoj Kumar in 2022 [11] used Optical Character Recognition (OCR) to recognize easily identifiable handwritten characters on documents. This study proposed the J&M model for text detection from handwritten images.

Further research by Asghar Ali Chandio, Md. Asikuzzaman, Mark Pickering, and Mehwish Leghari in 2020 [5] focused on the natural recognition of Urdu text from images that were presented and analyzed. This study utilized 2500 images of natural scenes captured using digital cameras and built-in mobile phone cameras.

Another study by Tobias Schlagenhauf, Markus Netzer, and Jan Hilinger in 2023 [2] discussed text detection on images independently. Detection techniques on images are a crucial step towards autonomous production machinery, especially for brown-field processes where no closed CAD-CAM solutions are available.

Lastly, research conducted by R.M Syachrul M.A.K, Moch Arif Bijaksana, and Arief Fathul Huda in 2019 [10] employed Named Entity Recognition (NER), which is an information extraction technique aimed at detecting entities such as people's names, locations, events, and times more quickly.

### 2.2. Data Collection

The data used in this study consists of image data. Each word from the Bugis and Makassar languages is photographed. The data is collected from handwritten texts. Below is an example of the data used in the research.

<table>
<thead>
<tr>
<th>Jenis Lontara</th>
<th>Kata Bugis</th>
<th>Kata Makassar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ello</td>
<td>Allo</td>
<td></td>
</tr>
<tr>
<td>Alusu</td>
<td>Alusu’</td>
<td></td>
</tr>
<tr>
<td>Bandera</td>
<td>Bandera</td>
<td></td>
</tr>
</tbody>
</table>

**Table 2. Lontara Bugis and Makassar Letter Data**
2.3 Preprocessing

The preprocessing stage consists of several parts. First, the image size is changed to 80 x 80 pixels. Next, the image data is processed for feature extraction. The step taken is to convert the color into a grayscale, commonly known as grayscale. Shape features are properties of an object that form a set of lines and contours. Formative features are classified according to the technology used. Classes are based on boundaries and areas. Boundary-based techniques describe the shape of regions using external features, such as pixels on the object's boundary [12].

2.4 Algoritma Viterbi

The Viterbi algorithm is a model from the Hidden Markov Model (HMM), where HMM is a statistical model that posits a system can be modeled as a Markov process with observable states.[13]. The Viterbi algorithm uses a Viterbi trellis to find the most optimal sequence of Hidden States for a given observable problem. To determine the optimal state sequence ( \( Q^{\ast} = \{ Q_1^{\ast}, Q_2^{\ast}, \ldots, Q_n^{\ast} \} \) ) with the observation sequence ( \( O = \{ O_1, O_2, \ldots, O_n \} \) ), the Viterbi algorithm is employed [14]. In the Viterbi algorithm, two additional variables are utilized. The Viterbi trellis involves recursive or backward calculations. Assuming ( \( O_t \) ) and ( \( Q_t \) ) as variables, the Viterbi trellis is calculated using Equation 1. In this case, the Viterbi trellis at time ( \( t \) ) for state ( \( i \) ) is denoted as ( \( V_t(i) \) ).

\[
V_t(i) = \max_{1 \leq j \leq N-1} V_{t-1}(j) A_{ij}(O_t)
\]

Keterangan dari Persamaan 1:
- \( V_{t-1}(i) \), Weathering Viterbi Trellis during Conditions t-1
- \( A_{ij} \), i.e. transition opportunities from \( O_t \) through \( Q_2 \)
- \( (O_t) \), is an emissions opportunity when \( O_t \) through \( t \)

The Viterbi algorithm aims to find the optimal estimation for the sequence of hidden states within an HMM, conditioned on a series of system measurements. At each stage, the Viterbi algorithm identifies the optimal value for states in the sequence and continues the analysis to the next stage in an inductive manner. [15].

3. Result and Discussion

This research uses python programming language. This study used image data processed in the preprocessing stage described earlier. The data is converted from lontara language into Latin words used in the viterbi process. The viterbi algorithm will test the data from the sentence inputted into it

Text classification using the Viterbi algorithm used the sentence "Anrengngi nara busi, annangngi nara batari". One example of a sentence used in testing the algorithm used.

Here's a table of each step in the viterbi algorithm:

Function ViterbiAlgorithm(Observations, States, TransitionProbabilities, EmissionProbabilities):
- Initialize Viterbi matrix \( V \) with size [len(States), len(Observations)]
- Initialize Backpointer matrix \( B \) with size [len(States), len(Observations)]
- # Initialization step
  - for each state in States:
    - \( V[0][0] = \text{initial probability of state} \) * emission probability of state given observation[0]
    - \( B[0][0] = 0 \) # Base case, no backpointer for initial step
- # Recursion step
  - for \( t \) from 1 to len(Observations) - 1:
    - for current_state in States:
      - max_probability = 0
      - max_previous_state = None
      - for previous_state in States:
        - probability = \( V[previous_state][t-1] \) * TransitionProbabilities[previous_state][current_state] * EmissionProbabilities[current_state][Observations[t]]
      - if probability > max_probability:
max_probability = probability
max_previous_state = previous_state
V[current_state][t] = max_probability
B[current_state][t] = max_previous_state

# Termination step
best_path_probability = 0
best_final_state = None
for each state in States:
    if V[state][len(Observations) - 1] > best_path_probability:
        best_path_probability = V[state][len(Observations) - 1]
        best_final_state = state

# Backtrace to find the best path
best_path = [best_final_state]
current_state = best_final_state
for t from len(Observations) - 1 down to 1:
    current_state = B[current_state][t]
    best_path.insert(0, current_state)
return best_path

# Define the input sentence
Observations = ["Anrengngi", "nara", "busi,", "annangngi", "nara", "batari"]

# Define the possible states
States = ["Lontara Bugis", "Makassar"]

# Define transition probabilities based on training data
TransitionProbabilities = {
    "Lontara Bugis": {"Lontara Bugis": 0.7, 
    "Makassar": 0.3},

    "Makassar": {"Lontara Bugis": 0.4, 
    "Makassar": 0.6}
}

# Define emission probabilities based on training data
EmissionProbabilities = {
    "Lontara Bugis": {"Anrengngi": 0.2, "nara": 0.5, "busi,": 0.3, "annangngi": 0.1, "batari": 0.4},
    "Makassar": {"Anrengngi": 0.3, "nara": 0.4, 
    "busi,": 0.6, "annangngi": 0.9, "batari": 0.6}
}

# Run Viterbi Algorithm
best_path = ViterbiAlgorithm(Observations, States, TransitionProbabilities, EmissionProbabilities)
print("Best path:", best_path)

In the pseudocode mentioned above the observation is the sentence "Anrengngi nara busi, annangngi nara batari is classified. Furthermore, state was identified as a classification label in Bugis and Makassar languages. The next step defines transition probabilities and emission probabilities based on the given training data. Next run the Viterbi algorithm using Observations, states transition probabilities and predefined emission probabilities. The result is the best route chosen by the algorithm that will include a classification label for each word in the sentence.

Here are the Emissions, transitions and probability results generated in the test.

Table 3. Emissi

<table>
<thead>
<tr>
<th>Kata</th>
<th>Bugis</th>
<th>Makassar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anrengngi</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>Nara</td>
<td>0.1</td>
<td>0.9</td>
</tr>
<tr>
<td>Busi</td>
<td>0.8</td>
<td>0.2</td>
</tr>
<tr>
<td>Annangngi</td>
<td>0.2</td>
<td>0.8</td>
</tr>
<tr>
<td>Batari</td>
<td>0.9</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 4. Transisi

<table>
<thead>
<tr>
<th></th>
<th>Bugis</th>
<th>Makassar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>Makassar</td>
<td>0.6</td>
<td>0.4</td>
</tr>
<tr>
<td>Bugis</td>
<td>0.3</td>
<td>0.7</td>
</tr>
</tbody>
</table>
Next the probability for each possible class sequence is based on the sentence "Anrengngi nara busi, annangngi nara batari." using the Viterbi algorithm:

<table>
<thead>
<tr>
<th>Kata</th>
<th>Bugis</th>
<th>Makassar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anrengngi</td>
<td>0.45</td>
<td>0.05</td>
</tr>
<tr>
<td>Nara</td>
<td>0.027</td>
<td>0.162</td>
</tr>
<tr>
<td>Busi</td>
<td>0.0388</td>
<td>0.02268</td>
</tr>
<tr>
<td>Annangngi</td>
<td>0.0046656</td>
<td>0.0126816</td>
</tr>
<tr>
<td>Batari</td>
<td>False</td>
<td>0.00005925232</td>
</tr>
</tbody>
</table>

The word "anrengngi" indicates a probability result of 0.45 meaning to the Bugis language. The word "nara" is more inclined to Makassar. Likewise with the words "spark plug" and "annangngi". The word "batari" probability value is recognized in Makassar. In testing, the word "batari" is more often used for people's names in a few sentences tested.

The probability value for the Makassar class is higher than the Bugis class for the sentences tested Anrengngi nara busi, annangngi nara batari. Have greater value. The sentences tested showed a larger vocabulary of Makassarese. Every word that a sentence makes comes from the letter Lontara. The results of probability can be seen in table 5.

4. Conclusion
Based on the results of the research conducted, it was concluded that the use of viterbi algorithms in the classification of Makassar Bugis lontara texts obtained greater probability results in the Makassar language. Object recognition using image input is less than optimal in recognition. There are several things that need to be considered for further research. The more data used, the better the classification results. Future research can use better algorithmic methods such as classification methods used in general. More data is used to get better accuracy results.

5. Acknowledgment
The author congratulates the Ministry of Education and Culture, Research and Technology for being financially supported to carry out research for novice lecturers.

Reference


