bAI-bAI: A Context-Aware Transliteration System for Baybayin Scripts

Jacob Simon Bernardo and **Maria Regina Justina E. Estuar**
Ateneo Social Computing Science Lab
Ateneo de Manila University
Quezon City, Philippines
jacob.bernardo@student.ateneo.edu, restuar@ateneo.edu

Abstract

Baybayin, a pre-colonial writing system from the Philippines, has seen a resurgence in recent years. Existing studies on Baybayin OCR face challenges with ambiguous Baybayin words that have multiple possible transliterations. This study introduces a disambiguation technique that employs word embeddings (WE) for contextual analysis and uses part-of-speech (POS) tagging as an initial filtering step. This approach is compared with an LLM method that prompts GPT-4o mini to determine the most appropriate transliteration given a sentence input. The proposed disambiguation process is integrated into existing Baybayin OCR systems to develop **bAI-bAI**, a context-aware Baybayin transliteration system capable of handling ambiguous words. Results show that incorporating POS as a filter does not significantly affect performance. The WE-Only method yields an accuracy of 77.46% and takes 5.35*ms* to process one sample while leveraging GPT-4o mini peaks at a higher accuracy of 90.52% but with a much longer runtime of 3280*ms* per sample. These findings present an opportunity to further explore and improve NLP approaches in disambiguation methods.

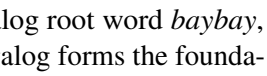
1 Introduction

Baybayin is an ancient writing system widely used by the early Filipinos starting around the 16th century (Solon, 2022; Lopez, 2021). The term *baybayin* comes from the Tagalog root word *baybay*, which means "to spell." Tagalog forms the foundation of the Filipino language (Santori, 2023).

The existence of Baybayin signifies the development of Philippine society prior to Spanish colonization (Lu, 2023). Baybayin provides a cultural and national emblem to the Filipino people and offers a sense of communal pride, belongingness, and social cohesion among diverse communities of Filipinos in the Philippines and abroad (Camba, 2021). Delving into Baybayin contributes to the conservation and comprehension of traditional Filipino practices, beliefs, and customs.

Recently, there has been a resurgence of interest in Baybayin that serves as fuel to cultural revival and artistic expression. Contemporary artists and writers have incorporated traditional elements into their work, as evidenced by the adoption of Baybayin across various creative mediums from paintings to tattoos (Narra Studio, 2019). Moreover, Baybayin has found its way into public spaces such as the Lagusnilad underpass and public transit systems (Balbutin Jr., 2023; Pornel, 2019). Legislative efforts have also been made to recognize Baybayin, such as the "National Writing System Act" by the House of Representatives which sought to declare Baybayin as the country's national writing system (Press and Public Affairs Bureau, 2018). In line with this, the Department of Education (DepEd) and the Commission on Higher Education (CHED) of the Philippines have incorporated Baybayin into their educational programs (Carasi, 2023).

While studies have been conducted on developing optical character recognition (OCR) systems that can transliterate Baybayin text, these systems lack awareness on the context surrounding the words. This becomes problematic in the presence of words that have ambiguous translations. Behind this gap arises the need for a context-aware model that is capable of identifying the appropriate word given multiple possible translations.

For example,  could mean either *"Bawat isa ay may **boto** sa halalan"* (as in, *"Everyone has a **vote** in the elections"*) or *"Bawat isa ay may **buto** sa halalan"* (*"Everyone has a **bone** in the elections"*). Given the electoral context of the sentence, it is more appropriate to choose the term "boto" ("vote") as the correct transliteration.

This study presents bAI-bAI, a context-aware system that is able to identify the correct transliteration of ambiguous Baybayin words. It tests and

Proceedings of the Second Workshop in South East Asian Language Processing, pages 1–9

compares two methods in achieving this goal: (1) a more traditional yet more explainable approach that uses word embeddings in conjunction with a part-of-speech tagger; and (2) a more advanced yet more blackbox-like approach that queries an LLM API.

2 Related Literature

2.1 The Baybayin writing system

Baybayin is written and read from left to right then top to bottom. It is an abugida or alphasyllabary, meaning that each character represents a consonant with an inherent vowel sound. In the case of Baybayin, this inherent sound is /a/.

Figure 1: The Baybayin alphabet

Each character represents either a vowel or a consonant-vowel combination, as illustrated in Figure 1; and words are separated by a space, with no explicit punctuation marks used in traditional Baybayin texts. Context and word familiarity often guide readers in deciphering the intended meaning of a passage.

The Baybayin script consists of 17 characters, each representing a specific sound in the Tagalog language. These characters are organized into three groups: patinig (vowels), katinig (consonants), and kudlit (diacritics).

The patinig characters represent the Baybayin vowel sounds [a], [e/i], and [o/u]. Unlike katinig characters, these patinig ones do not undergo modification by kudlit marks.

The katinig characters, on the other hand, represent the consonant sounds [ba], [ka], [da/ra], [ga], [ha], [la], [ma], [na], [nga], [pa], [sa], [ta], [wa], and [ya]. The kudlit marks, when placed above or below a character, indicate a change in pronunciation. For instance, a kudlit placed above a consonant character changes its inherent vowel sound from /a/ to /e/ or /i/. Written below, the vowel sound changes to /o/ or /u/. Additionally, a krus-kudlit (cross mark) written under a character eliminates the vowel /a/ after the consonant.

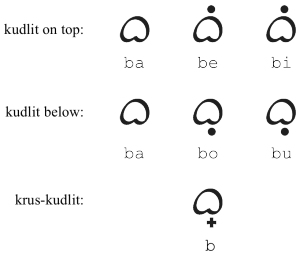


Figure 2: Usage of the kudlit (diacritic)

2.2 Current Baybayin transliteration systems

Research on automated Baybayin transliteration is still in its infancy stage (Pino et al., 2021b). Upon examination of existing studies, the end-to-end process tends to involve two fundamental steps: character-level classification, then word-level transliteration. Some papers focus purely on classification while others include a proposal of their own methods of transliterating a Baybayin word.

2.2.1 Character classification

Character classification serves as the foundation of the transliteration process. The classes are the Latin alphabet equivalents of the Baybayin symbols.

Ligsay et al. (2022) used a convolutional neural networks (CNN) based on YOLOv3 that reached a 98.92% accuracy. Amoguis et al. (2023) also implemented a CNN architecture and trained their model on an expanded dataset, resulting to an F1 score of 85.84%. Moreover, two other papers used CNNs and reported accuracy scores of 97.62% and 97.40% (Oraño et al., 2022; Vilvar et al., 2022). Beyond neural networks, Pino et al. (2021b) developed a support vector machine that yielded a 95.80% accuracy and 95.62% F1 score.

2.2.2 Word transliteration

In word transliteration, the output is essentially the combination of individual character classifications. A Baybayin word is segmented into singular symbols, each of which is converted to its Latin equivalent. Then, these converted characters are concatenated to form the final transliteration.

Two studies both used a Filipino corpus in executing this step, but they are distinct in how they determine and display the final output. Pino et al. (2021a) focused on the possibility of multiple transliterations as a result of the same symbols for

the following pairs of characters: [e/i], [o/u], and [da/ra]. To account for this, they carried out an alteration process in which all possible transliterations are found by considering both sounds that can correspond to the same symbol or diacritic. Take for example the Baybayin word ᜑᜒᜆᜓ , the alteration process would output the strings "heto," "hetu," "hito," and "hitu." The final ones, then, that would be displayed are the valid words that have a match in the corpus, which in this case are "heto" ("here") and "hito" ("catfish").

Alternatively, Vilvar et al. (2022) used the Levenshtein distance between the predicted word and the words in the corpus to determine the output. If the predicted word is in the corpus (i.e., the distance is zero), the output will be the predicted word. If it is not, the model will state that the predicted word is not in the corpus and display the corpus word with the shortest distance.

2.2.3 Script detection

Recognizing that some images may contain a mix of Latin and Baybayin characters, Pino et al. (2022) introduced a new first step to the transliteration process: script detection. The system first identifies whether a word is written in Latin or in Baybayin before proceeding on to the existing steps of classification and transliteration. If the word is in Latin already, there would be no need to transliterate.

This was implemented by first classifying each character to either Baybayin or Latin; then, the final prediction of the word script would be whichever classification is more dominant. For example, if three out of five characters in a word was determined to be Baybayin, then the entire word will be classified as Baybayin. With this proposed step, the resulting transliteration system achieved an accuracy of 93.64%.

2.2.4 Challenges and limitations

Although OCR and transliteration systems have been developed for Baybayin, all of them classify symbols with diacritics as only either consonant-[e/i] or consonant-[o/u] (Amoguis et al., 2023; Ligsay et al., 2022; Pino et al., 2022; Oraño et al., 2022; Vilvar et al., 2022). There exists no literature yet on a translator that can determine whether a word should have an /i/ instead of an /e/ (and vice versa) or an /u/ instead of an /o/ (and vice versa).

Moreover, other challenges in the process involve accurately detecting and classifying the kudlit, or diacritics, such as identifying the [o/u] diacritic (see *kudlit below* in Figure 2) as distinct from a cross mark (see *krus-kudlit* in Figure 2). The kudlit also tends to be detected as separate from the main character, resulting in incorrect or invalid transliterations. On the other hand, limitations are generally concerned with the clarity of how the Baybayin symbol is written; and most existing systems only implement one-word or even character-level transliteration and do not have the capability to handle phrase or sentence blocks (Amoguis et al., 2023; Ligsay et al., 2022; Oraño et al., 2022; Vilvar et al., 2022).

2.3 Transliteration systems for other non-Latin scripts

The available literature on NLP systems that have the capability of handling multiple transliterations based on context and meaning is remarkably limited. Even beyond Baybayin, only two published papers appear to have a similar goal of resolving such ambiguities.

2.3.1 From Shahmukhi to Gurmukhi

In 2011, Saini and Lehal (2011) proposed two methods for disambiguating words when transliterating from Shahmukhi to Gurmukhi, two different writing systems for the Punjabi language. The first algorithm uses as state sequence representation as a Hidden Markov Model (HMM) while the second approach proposes an n-gram model with a context window size of ± 5. Both statistical approaches achieved an accuracy score of more than 92%.

2.3.2 From English to Cyrillic

In 2014, Spasov (2014) designed an algorithm to handle ambiguity when translating full sentences from English to Cyrillic, a script used for various languages in Eurasia. The algorithm focused on words with dual meanings arising from non-standard transliteration circumstances and grammar rules. For example, the word "zabar" as written with the English alphabet could translate to either забар or жабар in Cyrillic. The transliteration process relies heavily word co-occurrence frequency within sentences from the corpus: the word that appears more frequently with other words in the sentence is chosen as the correct transliteration.

3 Methodology

The end goal of this study is to develop **bAI-bAI**, an end-to-end context-aware system that can

transliterate Baybayin script, including ambiguous words, to Filipino given an input image. This process entails four main stages: acquisition and preprocessing of data, development of the two word disambiguation systems, evaluation of both systems, and finally, integration of the disambiguation process with an OCR model.

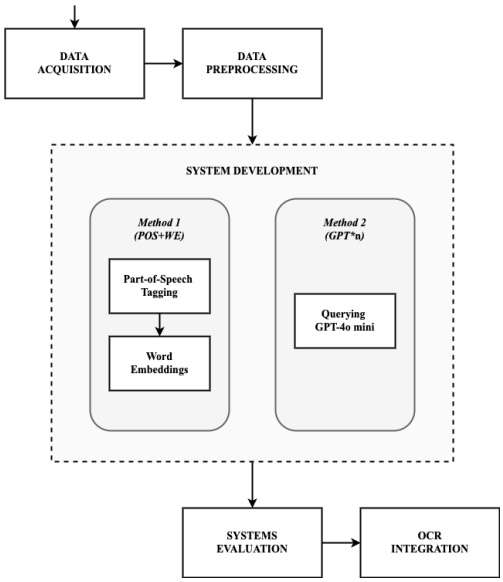


Figure 3: Flowchart of methodology

3.1 Data acquisition and preprocessing

3.1.1 Corpus

The Palito Tagalog corpus developed by Dita et al. (2009) was expanded with a collection of short stories written in Filipino to build the corpus for this study, with the final corpus having 627,711 words in total, of which 35,474 are unique. In preparing the corpus for generating word embeddings, several preprocessing steps were implemented. First, accented characters were replaced with their unaccented counterparts and all characters were converted to lowercase to ensure uniformity in representation. Then, punctuation marks, numerical digits, and other non-alphabetic characters were removed from the text. Finally, stopwords, as compiled by Diaz Jr. (2016), were excluded from the corpus.

3.1.2 Dictionary

The UP Diksiyonaryong Filipino was acquired through web scraping of the `diksiyonaryo.ph` platform, where only words with letters that can be represented in the Baybayin alphabet were retrieved. These letters are [a], [b], [k], [d], [r], [e], [i], [g], [h], [l], [m], [n], [ng], [p], [s], [t], [o], [u], [w], [y]. Each entry in the dictionary contained four pieces of information: (1) the word itself, (2) its definition, (3) its parts of speech, and (4) its language or dialect of origin. The resulting dictionary was structured in the JSON format.

The dictionary encompasses 57,225 words, with each word averaging two definitions and each definition consisting of approximately six words. The words were grouped into 11 distinct parts of speech, namely, pangngalan (noun), panghalip (pronoun), pang-uri (adjective), pandiwa (verb), pang-abay (adverb), pangatnig (conjunction), pang-ukol (adposition), pantukoy (article), padamdam (interjection), panlapi (affix), and pang-angkop (ligature).

3.1.3 Sample selection

A total of 1,058 text blocks containing one ambiguous word were randomly selected from the corpus. The test sample size was limited to this number primarily due to the constraints of available hardware resources during evaluation. Each text block contains a mean of 29 words and 159 characters; and each ambiguous word, on average, has two possible transliterations.

To determine if a text block contained an ambiguous word, the block was iterated over to identify if a word contained any of the following letters: ‘e’, ‘i’, ‘o’, ‘u’, ‘d’, ‘r.’ For each word that did, the word was altered so that all of its permutations were generated (i.e., e’s were replaced with i’s and vice versa, o’s were replaced with u’s and vice versa, and d’s were replaced with r’s and vice versa). Each word-permutation was then verified against the dictionary to determine if it is a valid word. Finally, all valid words were marked as possible transliterations; and all text blocks containing words with multiple possible transliterations were considered ambiguous.

Each data point in the sample set contained the following information: the text block, the correct word, and the possible transliterations.

Text blocks were also converted to Baybayin using an algorithm by Brennan (2020) to serve as sample inputs to bAI-bAI, which was designed to accept image inputs of Baybayin.

3.2 Development of word disambiguation systems

Two methods were implemented to disambiguate Baybayin words: (1) the more traditional yet more explainable approach: a combination of part-of-speech tagging and word embeddings manipulation

(POS/WE); and (2) the more advanced yet more blackbox-like approach: prompting GPT-4o mini (GPT*n).

3.2.1 Part-of-speech tagger & word embeddings

Part-of-speech (POS) tagging is a technique that involves assigning grammatical categories—or, parts of speech such as noun, verb, or adjective—to each word in a sentence (Pykes, 2020). This study used a Tagalog POS tagger from the `calamanCy` library, an NLP preprocessing framework for Tagalog, developed by Miranda (2023). From 13 parts of speech that can be detected by this tagger, the following tags had equivalent parts of speech in the scraped dictionary: noun, pronoun, adjective, verb, adverb, conjunction, adposition, and interjection.

The POS tagger served as a filtering mechanism to determine if the word embeddings step was necessary. If the category identified by the POS tagger matched only that of one word in the possible transliterations, then that word was chosen. For example, consider the sentence *"Gusto ko pumunta sa* ᜒᜎᜓ *"* (*"I want to go to* ᜒᜎᜓ*"*). Here, the word ᜒᜎᜓ could represent either "peru" ("Peru," referring to the country; noun) or "pero" ("but;" conjunction). The POS tagger detects that the word where the Baybayin word appears should function as a noun. Since only "peru" corresponds to the noun attribute, that word is automatically selected. In the case where multiple words or none match the target category, the method proceeds on to utilize word embeddings.

Word embeddings are a method for representing words in a continuous vector space, where the distance and direction between vectors denote the similarity and relationships among the represented words (Barnard, 2024). Each word is mapped to a vector such that similar words have vectors closer together in the embedding space. The GloVe algorithm, an unsupervised learning technique, was implemented to generate word embeddings with a dimensionality of 50 for all words in the corpus.

The resulting word embeddings were used to determine which word definition—among the definitions of the possible transliterations—was most similar to the meaning of the input sentence. Then, the word with the definition that was semantically closest to the input sentence was chosen. In other words, understanding the meaning of a word helps in determining which choice best fits the context of a sentence. To achieve this, sentence embeddings were computed as the average vector of all words in each text block (i.e., the input and the definitions); and semantic similarities were measured by calculating the cosine similarities between the sentence embeddings of the input and each definition.

In essence, the POS module narrows down word choices based on grammar; for instance, if the Baybayin word in context is expected to be a verb, then only verb candidates are considered. If this step alone cannot resolve the ambiguity, word embeddings are utilized to select the word that is most meaningful within the sentence context. This combination is labeled as the **POS+WE** approach. In addition to POS+WE, the word disambiguation process was also tested using word embeddings only (**WE-Only**), without an initial filtering by the POS tagger.

3.2.2 GPT-4o mini

GPT-4o mini is a version of OpenAI's Generative Pre-trained Transformer (GPT) series, a family of large language models designed to understand and generate human-like text based on the input it receives (OpenAI, 2024). Utilizing GPT for Baybayin word disambiguation requires querying with an input prompt.

First, the system role was defined with the following directive: `You are a linguistic expert specializing in the Filipino language.` Then, with an input prompt, the GPT model was asked to identify the most appropriate transliteration from a set of options based on the provided context.

Consider the sentence *"* ᜐᜒᜆᜓ᜔ *mo sa kanya kung paano ang magluto"* ("ᜐᜒᜆᜓ᜔ them how to cook"). The possible transliterations for ᜐᜒᜆᜓ᜔ are "itodo" (to give it one's all) and "ituro" (to teach). In this case, the prompt is formulated as follows:

`'_____ mo sa kanya kung paano ang magluto' Which is more appropriate to fill in the blank: 'itodo' or 'ituro'? Respond with strictly just the word.`

Given the potential inconsistency of the responses of GPT, each prompt was passed for several iterations; and the word with the most occurrences was selected. For example, with five iterations, the responses may be ["itodo", "ituro", "ituro", "itodo", "ituro"]. Since the word "ituro" appears more frequently (3 out of 5 responses), it is chosen as the appropriate transliteration. The output sentence shall then be *"ituro mo sa kanya*

kung paano ang magluto" ("teach them how to cook"). The performance of this method was evaluated across 1-, 3-, 5-, 7-, and 9-iteration approaches. These are referred to as **GPT*n**.

3.3 Systems evaluation

The performance of both systems were evaluated using two metrics: accuracy and runtime. Accuracy was calculated as the ratio of correctly identified words to the total number of valid input samples, measuring effectiveness. On the other hand, runtimes for individual sample processing were recorded to quantify the efficiency of the systems.

In this problem, accuracy was chosen as the performance metric because the errors are strictly binary—either the selected transliteration matches the correct word or it does not. Unlike classification problems where errors can be categorized as false positives or false negatives, this task involves string matching where the output should match the correct word exactly, with no variations in spelling.

To determine if the performances of each approach were significantly different, a t-test was conducted to compare POS+WE with WE-Only, while the One-Way ANOVA was performed for the various n-iterations of GPT. Barlett's test was initially carried out to confirm the equality of variances across the GPT iterations. Then, the best-performing approaches using POS and/or WE and using GPT were identified and further compared using a t-test.

3.4 Integration with OCR model

The development of bAI-bAI involved the integration of the proposed disambiguation process with an existing CNN-based Baybayin OCR model designed by Vilvar et al. (2022) (refer to Chapter 2.2). Challenges encountered with employing this model included difficulties in accurately detecting kudlit marks and limitations that do not allow beyond word-level transliteration.

3.4.1 Improving kudlit detection

The OCR model frequently misinterpreted kudlit marks as independent characters rather than as modifiers attached to core characters. To address this, the algorithm was modified to treat objects that are vertically aligned—based on their x-coordinates—as a single character. For instance, as demonstrated in Figure 4, when the bounding box for a detected object (the diacritic) is within the side edges of the bounding box of the core character, these two objects are merged together and recognized as one character.

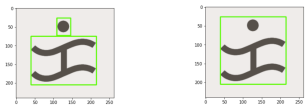


Figure 4: Kudlit detection before (left) and after (right) bounding box adjustment

A notable limitation of this approach is that it restricts bAI-bAI to only process Baybayin text along a single horizontal line. Since this algorithm merges characters based solely on their x-positions and ignores the y-positions, multi-line inputs are not supported.

3.4.2 Implementing sentence-level transliteration

The system proposed by Vilvar et al. (2022) was limited to transliteration at the word level as it did not have the functionality to detect word boundaries. To bridge this gap, the horizontal dilation value used for image input preprocessing was modified.

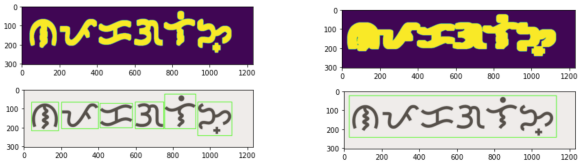


Figure 5: Original dilation (left) and modified horizontal dilation (right)

As shown in Figure 5, horizontal dilation was increased to connect individual characters. This enables the sequence of characters to be detected as a single contiguous object and therefore be recognized as one word. For each detected word block, the original algorithm for word-level transliteration is applied, and the resulting words are concatenated, separated by a space, to form the output sentence.

With this, an overview for the end-to-end process for transliterating a Baybayin text block is outlined in Figure 6. The resulting system, named bAI-bAI, consists of two primary processes: the OCR Transliteration step followed by the Word Disambiguation step.

4 Results and Discussion

The proposed disambiguation processes were applied to 1,058 text blocks containing words that are ambiguous when transliterated from Baybayin.

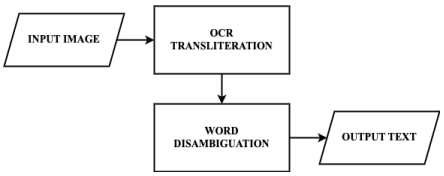


Figure 6: Flowchart of bAI-bAI pipeline

Method	Correct Samples	Accuracy
POS + WE	752/984	76.42%
(POS)	*(67/96)*	*(69.79%)*
(WE)	*(685/888)*	*(77.14%)*
WE-Only	749/967	77.46%
GPT*1	932/1058	88.09%
GPT*3	943/1058	89.13%
GPT*5	958/1058	90.55%
GPT*7	955/1058	90.26%
GPT*9	953/1058	90.08%
Random	*523/1058*	*47.65%*

Table 1: Resulting accuracies per method

As shown in Table 1, the highest accuracy was achieved by prompting GPT five times to determine the most appropriate transliteration. This improvement over its one-iteration approach suggests the benefits of prompt repetition. However, higher iterations did not provide significant improvements, and results for conducting seven and nine iterations even showed a slight decrease in performance. Although these observed accuracies might indicate some performance disparity, it is crucial to emphasize that statistical analysis reveals no significant difference in performance across the different iteration counts. This is presented in Table 2 and further elaborated upon in the subsequent discussion.

The POS+WE hybrid model yielded a 76.42% accuracy, with its individual components achieving 69.79% and 79.14%, respectively. While lower than GPT, POS+WE substantially outperforms the random guess baseline of 47.64%. Furthermore, although the WE-Only technique achieved a slightly higher accuracy of 77.46%, it does not imply that the POS step is redundant as the hybrid model was able to successfully process a larger proportion of the sample size (984 vs. 967). Rather, it suggests that improving the performance of the POS tagger may enhance overall system performance.

It is important to note that both approaches fell short of the total sample size (1,058) due to inconclusive cases. This limitation arises from the use of word embeddings, which are limited by the vocabulary present in the training corpus. When words from the input sentence or the definitions of a word do not exist in the corpus (i.e., out of vocabulary, or OOV), the calculation for cosine similarity results in `None`. This leads to an undetermined outcome due to the absence of quantifiable similarities. In certain instances, this can be addressed by the POS tagger, as evidenced by the lower number of inconclusive samples in the hybrid approach. However, when both the word embeddings component and the POS tagger fail to provide a conclusive match, the overall system is incapable of processing the input. This limitation was not factored into the calculation of accuracies presented in Table 1; instead, the numbers of successfully processed samples were reported to evaluate system performance irrespective of OOV words.

Group	Statistic	p-value
POS+WE vs. WE-Only	-1.02	0.323
among GPT*n's	1.30	0.285
WE-Only vs. GPT*5	-10.66	3.31e-09

Table 2: Results of statistical analyses

Statistical analysis revealed that adding POS to WE did not significantly improve accuracy over the WE-only approach, as indicated by a large p-value of 0.323 (see Table 2). This suggests that POS integration may not be necessary for word disambiguation. Furthermore, a comparison of the various n iterations of GPT yield a p-value of 0.285, indicating no significant accuracy gain with additional iterations. Thus, prompt repetition does not appear to provide statistically meaningful advantages. Consequently, the approaches with the highest accuracies were selected to compare between the traditional and the blackbox methods—WE-Only and GPT*5. With a large t-score of -10.66 and a p-value of 3.31×10^{-9}, GPT*5 performs significantly better than WE-Only.

Besides accuracy, runtime performance is another key consideration in the comparison of these approaches. When considering larger datasets or real-time processing scenarios, faster methods can offer significant advantages. The execution times for processing individual samples are presented in Table 3.

Although GPT*5 performed best in terms of identifying the correct transliteration, its efficiency significantly lagged behind the traditional alternatives. Even the single-iteration version of the GPT

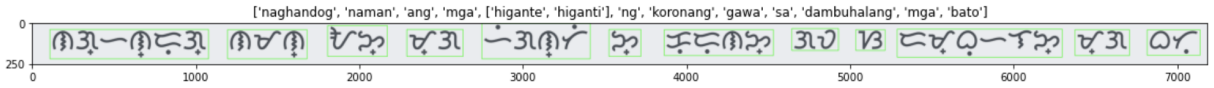


Figure 7: Sample output and visualization of OCR Transliteration step

Method	Runtime
POS + WE	27*ms*
WE-Only	5.35*ms*
GPT*1	670*ms*
GPT*3	2190*ms*
GPT*5	3280*ms*
GPT*7	5890*ms*
GPT*9	6120*ms*

Table 3: Runtimes per method for one sample

method operated around 25 times slower than the POS+WE technique. This disparity was amplified by the higher-iteration versions, with GPT*5 exceeding the processing time of POS+WE by a factor of over 121 and GPT*9 by a factor of 227. On the other hand, the WE-Only approach stood out for its remarkable speed, processing one sample in around 5*ms*.

With this disambiguation process integrated with the modified OCR model, the **bAI-bAI** system follows the pipeline illustrated in Figure 6.

A sample output of the OCR Transliteration step is presented in Figure 7 while Figure 8 details the corresponding output of the Word Disambiguation step.

```
naghandog naman ang mga _________ ng koronang gawa sa dambuhalang mga bato
Possible transliterations: ['higante', 'higanti']

Using POS tagger...
Ambiguous word is NOUN
UP Dictionary equivalent tag: png
POS matching failed: multiple matches.

Using word embeddings...
Cosine similarities for definitions of "higante"
0.52333814
0.22349443
0.32916453
Cosine similarities for definitions of "higanti"
-0.10025481
Chosen word: higante
Correct word: higante
```

Figure 8: Sample output of Word Disambiguation step using POS+WE

In Figure 7, the fifth Baybayin word exhibits ambiguity as indicated by the presence of two possible transliterations: “higante” (“giant”) and “higanti” (“revenge”), which were stored in a sublist. The sentence in English could translate to either *“the **giants**, in turn, offered a crown made of enormous gemstones”* or *“the **revenges**, in turn, offered a crown made of enormous gemstones”*. Following the OCR Transliteration step, the Word Disambiguation step determined the appropriate word to be “higante” (“giant”), which is the correct and contextually meaningful transliteration.

5 Conclusion

This study introduced and compared two methodologies for resolving ambiguities in Baybayin words with multiple possible transliterations: (1) the more traditional yet more explainable POS+WE method, and (2) the more blackbox-like yet more advanced GPT*n method. Empirical evaluation of these approaches showed that GPT*5 was the most effective with an accuracy of 90.52%. However, this was accompanied by a substantial increase in computational demand, with GPT*5 requiring 121 times more processing time than POS+WE. Despite its lower accuracy of 77.46%, WE-Only was considerably more efficient with its capability to process one sample in a mere 5.35*ms*. Furthermore, adding POS to WE and prompt repetition for GPT do not appear to provide statistically significant benefits.

These disambiguation techniques were integrated with a Baybayin OCR model to implement bAI-bAI, an end-to-end system capable of transliterating Baybayin text blocks, including those with ambiguous words.

6 Recommendations

Currently, no other simple transliteration systems for Baybayin exist in literature. Future work may explore alternative disambiguation methods by applying other natural language processing techniques for comparison. The performance of the POS+WE approach may also be improved by utilizing a more robust POS tagger and by expanding the training corpus and the dictionary. Finally, further research to improve the current state of Baybayin OCR would enhance overall system performance.

References

Adriel Isaiah V. Amoguis, Gian Joseph B. Madrid, Benito Miguel D. Flores IV, and Macario O. Cordel II. 2023. Baybayin character instance detection. *arXiv preprint arXiv:2304.09469*.

Albert Balbutin Jr. 2023. Baybayin in public places and shops.

Joel Barnard. 2024. What are word embeddings?

Fredrick R. Brennan. 2020. Advanced, correct Baybayin converter.

Allan Torres Camba. 2021. *Baybayin: The Role of a Written Language in the Cultural Identity and Socio-Psychological Well-Being of Filipinos*. Ph.D. thesis, Harvard University.

Ivy Carasi. 2023. Reviving baybayin: A collective effort to preserve filipino heritage.

Gene Diaz Jr. 2016. Stopwords Tagalog (TL).

Shirley N Dita, Rachel Edita O Roxas, and Paul Inventado. 2009. Building online corpora of philippine languages. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation*, pages 646–653. Waseda University.

Angel Mikaela P. Ligsay, John B. Rivera, and Jocelyn F. Villaverde. 2022. Optical character recognition of baybayin writing system using yolov3 algorithm. In *2022 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, pages 1–5. IEEE.

Jeanylyn Lopez. 2021. Baybayin: A writing script used in pre-Hispanic Philippines.

Brian James Lu. 2023. Preserving our cultural heritage: The revival of Baybayin.

Lester James Miranda. 2023. calamanCy: A Tagalog natural language processing toolkit. In *Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023)*, pages 1–7, Singapore, Singapore. Empirical Methods in Natural Language Processing.

Narra Studio. 2019. Baybayin: The ancient Filipino script lives on.

OpenAI. 2024. Models; Flagship models.

Jannie Fleur V Oraño, Marco Eraño P Pahamotang, and Rhoderick D Malangsa. 2022. Using deep learning and adaptive thresholding approach for image-based Baybayin to Tagalog word transliteration. In *2022 IEEE 14th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, pages 1–6. IEEE.

Rodney Pino, Renier Mendoza, and Rachelle Sambayan. 2021a. A Baybayin word recognition system. *PeerJ Computer Science*, 7:e596.

Rodney Pino, Renier Mendoza, and Rachelle Sambayan. 2021b. Optical character recognition system for baybayin scripts using support vector machine. *PeerJ Computer Science*, 7:e360.

Rodney B. Pino, Renier G. Mendoza, and Rachelle R. Sambayan. 2022. Block-level optical character recognition system for automatic transliterations of baybayin texts using support vector machine. *Philippine Journal of Science*, 151(1).

Atom L. Pornel. 2019. Baybayin and early filipino images greet riders on LRT-1.

Press and Public Affairs Bureau. 2018. House approves Baybayin as national writing system.

Kurtis Pykes. 2020. Part of speech tagging for beginners.

Tejinder Singh Saini and Gurpreet Singh Lehal. 2011. Word disambiguation in shahmukhi to gurmukhi transliteration. In *Proceedings of the 9th Workshop on Asian Language Resources*, pages 79–87.

Ann Santori. 2023. Tagalog language | History, alphabet misconceptions.

Kryshia Gayle Solon. 2022. How Baybayin's legacy lives on.

Stojance Spasov. 2014. *Web service for ambiguous transliteration of full sentences from Latin to Cyrillic alphabet*. Ph.D. thesis, University Goce Delcev-Stip.

Ric Andrei Vilvar, Daniel Shawn Ceballo Hammond, Francis Mark Ricohermoso Santos, and Hernan S. Alar. 2022. Baybayin script word recognition and transliteration using a convolutional neural network. *Available at SSRN 4004853*.