Comparing a Thai Words Segmentation Methods in the LST20 Dataset

Krittapol Damrongkamoltip[1], Khatcha Ruenlek[2], Wasit Limprasert[3,*], and Prachya Boonkwan[4,*]

Received: 26 March 2024;
Revised: 13 May 2024;
Accepted: 17 May 2024;
Published: 12 August 2024;

Abstract

In this era of globalization where information is widely available, organizations are increasingly placing importance on using information to enhance their business. Although data is easily available, there are still challenges in natural language processing tasks, especially, the division of Thai words that lacks clarity of word boundaries, etc. This makes it difficult to identify the word groups in a sentence appropriately. Therefore, this study focuses on evaluating the performance of the word segmentation method including the Dictionary use and learning from data using evaluation of word segmentation in six techniques are important goals for the verification of the literal level accuracy and processing time of each method and technique, by the LST20 dataset contains 3,745 documents and covers 15 news categories in results show a more efficient way to learn from data.

Keywords: Thai Words, Segmentation Methods, LST20 Dataset.

1. Introduction

In the age of social media explosion, data has become vital for organizations seeking to gain a competitive advantage and optimize management processes. Text data, prevalent across social platforms, fuels a process called Natural Language Processing (NLP) that unlocks valuable insights. NLP plays a critical role in a wide range of real-world applications, especially those involving user interaction. One of the major challenges in language processing is word segmentation, also known as word tokenization. This process involves breaking down written language into distinct units. While some languages, like English, Spanish, and French, benefit from clear word boundaries and relatively fixed structures, others like Chinese, Thai, and Vietnamese present complexities due to the absence of explicit word boundaries. In

[1] Student, Data Science and Innovation, College of Interdisciplinary Studies, Thammasat University, Pathum Thani 12121, Thailand; Email: krittapol.dam@dome.tu.ac.th

[2] Student, Data Science and Innovation, College of Interdisciplinary Studies, Thammasat University, Pathum Thani 12121, Thailand; Email: khatcha.rue@dome.tu.ac.th

[3,*] Assistant Professor, Dr., Data Science and Innovation, College of Interdisciplinary Studies, Thammasat University, Pathum Thani 12121, Thailand; Email: wasit@tu.ac.th

[4,*] Lecturer, Dr., Language and Semantic Technology Research Team, National Electronics and Computer Technology Center, Pathum Thani 12121, Thailand; Email: prachya.boonkwan@nectec.or.th

*Corresponding authors: Wasit Limprasert (wasit@tu.ac.th); Prachya Boonkwan (prachya.boonkwan@nectec.or.th)

Citation:
Damrongkamoltip, K., Ruenlek, K., Limprasert, W., & Boonkwan, P. (2024). Comparing a Thai Words Segmentation Methods in the LST20 Dataset. *Journal of Computer and Creative Technology, 2*(2), 61-70. https://doi.org/10.14456/jcct.2024.7.

these languages, words can span multiple syllables, and context often dictates where a word begins and ends. Thai words, for example, often lack clear delimiters and can encompass multiple syllables.

Accurate segmentation is paramount for NLP applications as it directly impacts tasks such as part-of-speech tagging, named entity recognition, and machine translation. Improved segmentation not only enhances the accuracy of these applications but also facilitates deeper linguistic analysis and more effective information retrieval. Therefore, understanding the complexities of Thai word segmentation and achieving precise segmentation are essential for advancing NLP research and applications in the Thai language. Various tools, ranging from dictionary-based methods to learning-based segmentation and sub word tokenization, offer solutions to address these challenges.

This study focuses on Thai language processing, aiming to evaluate the accuracy and efficiency of different segmentation methods. Our contribution lies in establishing a benchmark to determine the most suitable methods for diverse data contexts. We aim to provide valuable insights for selecting the most appropriate segmentation methods, ultimately reducing resource demands, and ensuring proper functionality across various data contexts. To achieve this, we compare six prevalent Thai word segmentation methods – Newmm, Colloc, Longest Matching, Attacut, Deepcut, and XLM-RoBERTa. We leverage the LST20 corpus (Boonkwan, 2020; National Electronics and Computer Technology Center, 2022) curated by Thailand's NECTEC, which comprises 3,745 annotated documents across 15 news genres. This extensive dataset provides a rich resource for in-depth exploration of Thai word segmentation methods.

2. Literature Review

Word segmentation in the Thai language has many challenges. For example, the lack of space delimiters, unlike other languages such as English, which has spaces between words, and the use of compound words, which are combinations of multiple words or syllables. (Haruechaiyasak et al., 2008) For example, the classic case "ตากลม" can be segmented in two different ways: "ตาก-ลม" (air dry) and "ตา-กลม" (round eyes) or "แม่น้ำ" (river), which is composed of "แม่" (mother) and "น้ำ" (water). Previous studies have explored different methodologies to tackle this issue, offering valuable insights into the evolution of approaches over time.

The existing literature on Thai word segmentation has provided valuable insights into different methodologies. For instance, a study titled "A Comparative Study on Thai Word Segmentation Approaches" conducted in 2008 compared dictionary-based (DCB) approaches to machine learning-based (MLB) approaches using n-gram transformations from the ORCHID corpus (Noyunsan et al., 2014). However, this study primarily focused on performance metrics, neglecting the crucial aspect of processing time, which is often a constraint in real-world applications with deadlines. Expanding on this, "A Multi-Aspect Comparison and Evaluation on Thai Word Segmentation Programs" in 2014 evaluated six Thai word segmentation programs: Libthai, Swath, Wordcut, CRF++, Thaisemantics, and Tlexs, using the BEST corpus 2014 developed by NECTEC (Aroonmanakun, 2002). This research not only assessed accuracy but also

Citation:
Damrongkamoltip, K., Ruenlek, K., Limprasert, W., & Boonkwan, P. (2024). Comparing a Thai Words Segmentation Methods in the LST20 Dataset. *Journal of Computer and Creative Technology, 2*(2), 61-70. https://doi.org/10.14456/jcct.2024.7.

considered processing time. Each program varied in its implementation environment, with differences in programming languages and support systems. While these studies contributed significantly to understanding Thai word segmentation, they also revealed certain limitations. For instance, each segmentation program required specific installations and environments, which could be cumbersome for users. Additionally, the rapid evolution of programming languages and frameworks has shifted developers' preferences towards specialization in a few languages rather than mastering multiple ones. To address these challenges and advance the field, our proposed research aims to compare various Thai word segmentation methods within a unified environment. By leveraging Python and utilizing the latest Thai language corpus, LST20, we intend to conduct a comprehensive comparison focusing on both accuracy and preprocessing time. This approach ensures consistency in evaluation metrics and simplifies the implementation process for users, aligning with contemporary trends in programming specialization and efficiency.

Regarding methodologies, two common approaches have been widely employed: the dictionary-based approach and the learning-based approach. The dictionary-based approach uses a dictionary to parse and segment text into words. This approach has a few popular selection methods, such as the longest-matching, which attempts to match a set of characters into the longest possible word (Poonwarawan, 1986). Another popular method is the maximum-matching, which finds every possible way to segment text and then selects the way with the fewest words (Virach, 1993). However, in this paper, we don't directly use the maximum-matching. Instead, we use the engine "newmm" from the library "PyThaiNLP." This engine makes use of the maximum-matching method and the Thai Character Cluster (TCC). The Thai Character Cluster uses the concept of a character cluster, which is a unit smaller than a syllable but larger than a character and matches them with defined rules (Theeramunkong et al., 2000). Additionally, we test the maximum collocation approach, which uses the idea of collocation strength between syllables to form words. Collocation can also be referred to as the co-occurrence of syllables, i.e., two syllables that are part of a word will have higher collocation strength. They use a dictionary to match all possible words from the syllables and then use collocation strength to select the best segmentation with the highest collocation strength.

However, advancements in machine learning have led to the learning-based approach, the word segmentation problem can be viewed as a binary-classification problem, trying to classify if a character is a word-beginning character (B) or a word-inning character (I). We compare three models for this problem: DeepCut (Kittinaradorn et al., 2019), AttaCut (Chormai et al., 2019), and an XLM-RoBERTa-based POS tagging model (De Vries et al., 2022). For the DeepCut model, they use a Deep Neural Network to perform the binary classification task. The Convolutional Neural network was trained with NECTEC's BEST corpus (Kittinaradorn et al., 2019). In the case of the AttaCut model, it's constructed using CNNs like the DeepCut model. However, there have been adjustments made to the convolutional layers, specifically focusing on minimizing overlap between them. As a result of this modification, the execution time is reduced compared to the DeepCut model. Additionally, AttaCut incorporates syllable knowledge through a

Citation:
Damrongkamoltip, K., Ruenlek, K., Limprasert, W., & Boonkwan, P. (2024). Comparing a Thai Words Segmentation Methods in the LST20 Dataset. *Journal of Computer and Creative Technology, 2*(2), 61-70. https://doi.org/10.14456/jcct.2024.7.

process known as syllable embedding, where characters within the same syllable share identical syllable embeddings (Chormai et al., 2019).

Lastly, the XLM-RoBERTa-based POS tagging model. The model we use is called XLM-RoBERTa base Universal Dependencies v2.8 POS tagging (De Vries et al., 2022). This model is fine-tuned from the pre-trained XLM-RoBERTa model (Conneau et al., 2020) for part-of-speech tagging using many different combinations of training and testing languages. The specific one we use was trained on the Hebrew language. As we mentioned, this model was originally trained for POS tagging, but it can also be used for word segmentation as it also has word boundaries as output.

3. Research Methodology

The experimental process consists of three phases: data preparation, data processing, and data measurement. The process commences with downloading the dataset through the openD portal (National Electronics and Computer Technology Center, 2022). Subsequently, data preprocessing is conducted to transform it into a usable format such as CSV using the DataFrame from the Pandas package. Concurrently, an environment for the experiment is set up, involving the download and installation of necessary tools, including Pip for acquiring essential packages and Python for running a script. After establishing the environment and completing the data preprocessing step, the second process consists of two parts, focusing on data measurement dimensions: accuracy and time processing. The transformed data is processed in a function implemented to work with six word segmentation methods: Newmm, Deepcut, Attacut, Longest, TLTK, and XML-Roberta. Almost all these methods are downloaded via Pip, except XML-Roberta, which is processed using an API pipeline implemented from Hugging Face. The results from each function are saved and exported in CSV format for use in subsequent stages. In the time processing stage, 300 records of mock-up data (De Vries et al., 2022) are used to examine the time processing of each word segmentation method. The mock-up data is divided into three equally sized datasets. Analogous to the accuracy processing, the data is passed through the word segmentation function. In addition to parsing the data before parsing the input data, the time is saved using the time module. Also, after parsing, the time is recorded as well. Finally, in the data measurement process, the results from the second process are compared with the ground truth answer. Accuracy is calculated using three metrics: Precision, Recall, and F1 score, employing a confusion matrix approach. Regarding the processing time is determined by finding the difference between the start and end parsing times.

5.1 Data Preparation

Our experiment began with the download of the LST20 corpus dataset from the openD portal (National Electronics and Computer Technology Center, 2022). Following the download, we set up the processing environment, which included the installation of Python and Pip. Five essential packages Pythainlp, Deepcut, Attacut, TLTK, and Request which needed to be downloaded to correspond with the selected word segmentations. We ensured that we installed the latest versions of all tools available at that time, namely Pythainlp v4.0.2, Deepcut v0.7.0, AttaCut v1.0.6, and TLTK v1.8.

Citation:
Damrongkamoltip, K., Ruenlek, K., Limprasert, W., & Boonkwan, P. (2024). Comparing a Thai Words Segmentation Methods in the LST20 Dataset. *Journal of Computer and Creative Technology, 2*(2), 61-70. https://doi.org/10.14456/jcct.2024.7.

Once the environment was configured, we imported the LST20 corpus from text files into a pandas dataframe which each file consists of many words that make up a sentence. In one row, there will be five components: word boundaries, POS tagging, named entities, clause boundaries, and sentence boundaries. If a word is a space, they use "_" as a symbol for space. This was achieved by utilizing the built-in function open() to open files from their path, retrieving content through the read() method, and then saving the content to dataseries and dataframe, respectively. The dataframe comprised two columns: the original sentences, concatenated from tokens in text files with join function, And the ground truth is representing the original data within the text files. To avoid processing errors, we consistently saved all outputs during the process as CSV files from the pandas dataframe, using the to_csv() function and setting the "encoding" parameter to "utf-8 - sig" to ensure readability when opened by other applications.

5.2 Data Processing

The processing began by importing the necessary packages mentioned above into the Python editor. We proceeded to tokenize the first column of original sentences until all sentences were processed, recording the start and end times for each sentence. This process is iterated in a loop for all selected word segmentations. During the parsing of sentences, the process unfolded as follows: first, we imported the necessary packages into the editor. Next, default values were set for each parameter of each tokenizer. For Attacut, the default model is "attacut-sc," which represents the best-performing model trained by the Attacut package. Regarding Deepcut, we utilize it directly from the Deepcut package by importing the tokenizer as follows: "from deepcut import tokenize as deepcut_tokenizerparameters," without specifying any additional parameters. Similarly, for TLTK, we employ it directly from the TLTK package using the following import statement: "from tltk.nlp import word_segment as tltk_tokenizer," where the word segmentation method utilized is based on the maximum collocation approach, denoted as 'colloc' by default. For both Newmm and Longest matching, we utilize them via the Pythainlp class named "from pythainlp.tokenize import word_tokenize as pythai_default_tokenizer" This class encompasses four parameters: engine, custom_dict, keep_whitespace, and join_broken_num. These parameters default to certain values, with custom_dict based on the Thai National Corpus (TNC) and keep_whitespace and join_broken_num set to True. The only parameter that differs among these methods is the engine parameter, which specifies the model or tokenizer object to be used. It is set to "newmm" for Newmm and "longest" for the longest matching approach. Before the iteration commenced, we need to create the list to store the output from the loop processed, assumed named as (y_tokenized_lst). The iteration commenced by taking a sentence from the first column of the imported dataframe or the full sentences from each file of LST20, stored in a temporary variable assumed named (X_text). Then, this temporary variable (X_text) was passed into the tokenizer, and the resulting tokenized output was stored in y_tokenized_lst. This process was repeated until all sentences from the first column were tokenized, and the entire process was replicated for other word segmentations.

Citation:
Damrongkamoltip, K., Ruenlek, K., Limprasert, W., & Boonkwan, P. (2024). Comparing a Thai Words Segmentation Methods in the LST20 Dataset. *Journal of Computer and Creative Technology, 2*(2), 61-70. https://doi.org/10.14456/jcct.2024.7.

And the last method XLM-Roberta model of sub word tokenization approach required different steps. First, we signed up for a Hugging Face account to create an API key token. Then, we imported the Request package into the editor and set variable values, including API_URL (the URL of the interested Hugging Face model API) and Headers ("Authorization": f"Bearer {'api_key'}"). Following this, we created a function for sending requests to the server using the POST method and set the POST parameters to "requests.post(API_URL, headers=headers, json=payload)" with payload representing the input data from the function. The output from the return method was set to JSON format for flexibility and ease of application. Subsequently, we executed the same process, passing the variable (x) storing sentences into the XLM-Roberta model function for tokenization. The resulting tokenized output was stored in another variable. Since the output from the XLM-Roberta model was in JSON format, we extracted only the required tokenized words from the JSON output using a loop and accessing the data value by the key named 'word.' These words were stored in a list, and the process was repeated until all sentences were tokenized.

Once the word segmentation was complete, we transformed the results from the tokenized process into a usable data type: a list of tokens. TLTK was the only word segmentation that returned results as strings but separated each word inside the sentence using "|". If it couldn't tokenize words, it returned the output as "<Fail>...</Fail>" or "\xa0 ," constraints that needed to be eliminated. After transforming the outputs, the next step was to calculate accuracy at the character-based level using the confusion matrix approach.

The confusion matrix serves as a table to evaluate the performance of a classification algorithm, employing four key elements: True Positive, False Positive, True Negative, and False Negative. These elements are variables in the formula for calculating metrics like precision, recall, and F1 score. Throughout this process, we conducted a character-based comparison between a list of tokenized tokens and the ground truth. Each character from both lists was scrutinized against one another, utilizing the last character of the token and its position to index the confusion matrix elements. To calculate accuracy, we transformed the last character from all tokens in the lists of all segments into new special characters absent in any of the sentences. Simultaneously, the last character of every token in the ground truth lists underwent a similar conversion. Following this conversion, all tokenized token elements in the list were concatenated into a sentence. Characters in the list of tokenized tokens were then classified based on their position in confusion matrix elements. The classification process encompassed storing the tokenized sentence and the ground truth sentence in variables, ensuring both sentences were of the same length. These sentences were then taken to a loop function to iterate through each character. Rules for classification were established: if a ground truth character matched a previously converted special character in the tokenized sentence, the True Positive value increased accordingly; if the ground truth character was a special character not matched in the tokenized sentence, the False Negative value increased; if the ground truth character was a different character and matched the position in the tokenized sentence, the True Negative value increased; for all other cases, the False Positive value

Citation:
Damrongkamoltip, K., Ruenlek, K., Limprasert, W., & Boonkwan, P. (2024). Comparing a Thai Words Segmentation Methods in the LST20 Dataset. *Journal of Computer and Creative Technology, 2*(2), 61-70. https://doi.org/10.14456/jcct.2024.7.

increased. This comprehensive process was repeated for every sentence from every word segmentation method, comparing them with the ground truth.

5.3 Time Processing

To measure the time processing, we randomly selected 300 sentences from a data frame containing LST20 data and divided them into three sets, each comprising 100 sentences. The execution process followed these steps: First, we imported the time package into the editor. Then, for each set of data, we brought it into the tokenizer and recorded the start time before tokenization using the time.time() function. Subsequently, we tokenized the sentences and recorded the end time after the words were cut, storing the time before and after parsing words separately in a list. We repeated this process until all sentences were tokenized and then repeated it again with other data sample sets. This entire procedure was continued until every word segmentation method had been thoroughly tested.

5.4 Data Measurement

Following the process of tokenization and subsequent tabulation of the elements constituting the confusion matrix, two pivotal dimensions were meticulously gauged: accuracy and time. In delving into accuracy, a trifecta of critical metrics—precision, recall (also known as sensitivity rate), and F1 - score—stood as the pillars of assessment. Each metric was calculated for every sentence, and the results were incorporated into the dataframe. The average was then calculated for each word segmentation using the mean function, providing the arithmetic average through the column. Regarding accuracy measurement, three metrics were used. Precision measures the accuracy of positive predictions made by the model, calculating the ratio of true positive predictions to the total number of positive predictions made by the model (both correct and incorrect). Recall measures the ability of the model to correctly identify all relevant instances, calculating the ratio of true positive predictions to the total number of actual positive instances in the dataset. F1 Score represents the harmonic mean of precision and recall, providing a balanced measure between the two metrics. When it came to measuring time, the process involved identifying the gap between when a sentence started and when it ended. This process aimed to determine the average time for each word breakdown, employing the same method used for accuracy assessment. By utilizing a mathematical mean function, which calculates an average, the total time for all word divisions was aggregated and then divided by the number of divisions, resulting in a number representing the average time taken.

4. Results and Conclusion

The results of link code and dataset were updated on the The Github website (https://github.com) (Ruenlek & Damrongkamoltip, 2023) and the results of comparing were as follows Table 1. and Table 2.

Citation:
Damrongkamoltip, K., Ruenlek, K., Limprasert, W., & Boonkwan, P. (2024). Comparing a Thai Words Segmentation Methods in the LST20 Dataset. *Journal of Computer and Creative Technology, 2*(2), 61-70. https://doi.org/10.14456/jcct.2024.7.

Table 1. A table showed the accuracy of six models across three metrics: precision, recall, and F1-score.

Measurement	Newmm	Colloc	Longest	Deepcut	Atta cut	xlm-roberta
Precision	**0.94**	**0.94**	0.12	**0.94**	0.93	0.13
Recall	0.85	0.89	0.31	0.81	**0.90**	0.27
F1 Score	0.89	0.91	0.17	0.87	**0.92**	0.17

Table 2. A table shows the processing time of six models across three sample dataset each 100 records.

Segmentation Methods	Processing Times			Average Processing Time
	1st Test	2nd Test	3rd Test	
Newmm	00.05	00.08	00.04	**00.05 s**
Colloc	101.47	30.13	21.90	51.17 s
Longest	12.74	330.98	06.61	116.78 s
Deepcut	04.47	02.70	01.68	02.95 s
Attacut	00.30	00.22	00.15	00.22 s

The results from the tables above demonstrate that both dictionary-based approach and learning-based approach have high accuracy methods and low accuracy methods. However, when considering processing time, the results above indicate that the learning-based approach processes faster than the dictionary-based approach.

4.1 Accuracy Dimension

Best Models (High Accuracy): Attacut and Colloc exhibit high accuracy across all three metrics—precision, recall, and F1-score. These models consistently achieve high values in all accuracy metrics, making them suitable choices for accurate word segmentation.

1) Precision: Deepcut, Attacut, and Newmm are the best, all scoring over 0.9.

2) Recall: Attacut stands out with 0.90, followed by Colloc (0.89) and xlm-roberta (0.27) is lowest recall score.

3) F1-Score: Attacut, and Colloc perform the best, each achieving over 0.90.

Average Models (Moderate Accuracy): Newmm and Deepcut perform reasonably well, achieving moderate accuracy levels across all metrics. While its performance is not as high as Attacut and Colloc, it still offers a decent balance between precision, recall, and F1-score.

Worst Models (Low Accuracy): Longest and xlm-roberta perform poorly in terms of accuracy. They have significantly lower precision, recall, and F1-score values compared to other models, indicating their inability to accurately segment words in the given context.

4.2 Processing Time Dimension

Fastest Models (Low Processing Time): Attacut, Newmm, and xlm-roberta are the fastest models in terms of processing time, with Newmm being the quickest overall. These models require very little time to process each sentence, making them suitable for applications where speed is a critical factor.

Citation:
Damrongkamoltip, K., Ruenlek, K., Limprasert, W., & Boonkwan, P. (2024). Comparing a Thai Words Segmentation Methods in the LST20 Dataset. *Journal of Computer and Creative Technology, 2*(2), 61-70. https://doi.org/10.14456/jcct.2024.7.

Moderate Processing Time: Deepcut has moderate processing times, with Deepcut being slightly faster. They strike a balance between accuracy and processing speed, making them viable options for applications where accuracy is essential, but speed is also a concern.

Slowest Model (High Processing Time): Longest and Colloc have a significantly longer processing time compared to other models, making it the slowest option. Its accuracy is also low, making it less preferable for most practical applications.

Best Model Selection: If the primary concern is accuracy and precision in word segmentation tasks, Attacut, Deepcut, and Newmm are the top choices respectively. They consistently outperform other models in accuracy metrics, making them suitable for applications where precise word boundaries are crucial. Consideration for Specific Use Cases: If the task requires very fast processing with a compromise on accuracy, Attacut and Newmm are the quickest options. However, it's essential to assess the specific use case and determine whether the sacrificed accuracy aligns with the application's requirements. Avoiding the Slowest and Least Accurate Model: Longest is both the slowest and least accurate option. Unless there is a specific use case where its characteristics align with the requirements, it is advisable to avoid using Longest due to its low accuracy and high processing time.

5. Discussion

The study explored the performance evaluation of various Thai word segmentation methods, shedding light on their accuracy and processing time. The findings underscore the importance of tailoring segmentation methods to specific use cases. For tasks prioritizing high accuracy, options like Attacut and Colloc emerge as frontrunners. Notably, learning-based approaches like Attacut or Deepcut demonstrate prowess in handling data containing new or informal words. Despite being trained mainly on formal data, their neural network-based architecture enables them to delineate word boundaries more effectively compared to segmentation methods based on dictionary approaches. Conversely, for swift processing, Attacut and Newmm stand out, albeit with a compromise on accuracy. However, when dealing with tasks involving formal data or requiring sentence segmentation into formal words, the dictionary-based approach emerges as a strong candidate. In this method, all words to be segmented are stored in a dictionary. Conversely, if the task involves detecting or identifying slang words or technical terms, this approach should be at the top of the list of potential solutions. This nuanced understanding enables practitioners to make informed decisions aligning with their application requirements.

Furthermore, the study prompts consideration of future avenues. Testing with diverse datasets, especially from platforms like TikTok and Twitter, offers a glimpse into real-world language complexities, encompassing slang, informal language, and varied topics. Exploring hybrid segmentation methods or integrating transformer models could potentially enhance accuracy and adaptability to evolving linguistic landscapes.

Citation:
Damrongkamoltip, K., Ruenlek, K., Limprasert, W., & Boonkwan, P. (2024). Comparing a Thai Words Segmentation Methods in the LST20 Dataset. *Journal of Computer and Creative Technology, 2*(2), 61-70. https://doi.org/10.14456/jcct.2024.7.

6. References

Aroonmanakun, W. (2002). Collocation and Thai Word Segmentation. *Proceedings of SNLP-Oriental COCOSDA*, 68-75. https://www.academia.edu/21349075/Collocation_and_Thai_Word_Segmentation.

Boonkwan, P., Luantangsrisuk, V., Phaholphinyo, S., Kriengket, K., Leenoi, D., Phrombut, C., Boriboon, M., Kosawat, K., & Supnithi, T. (2020). *The Annotation Guideline of LST20 Corpus*. ArXiv. https://doi.org/10.48550/ARXIV.2008.05055.

Chormai, P., Prasertsom, P., & Rutherford, A. (2019). *AttaCut: A Fast and Accurate Neural Thai Word Segmenter*. ArXiv. https://doi.org/10.48550/ARXIV.1911.07056.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2020). Unsupervised Cross-lingual Representation Learning at Scale. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 8440–8451. https://doi.org/10.18653/v1/2020.acl-main.747.

De Vries, W., Wieling, M., & Nissim, M. (2022). Make the Best of Cross-lingual Transfer: Evidence from POS Tagging with over 100 Languages. *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 7676–7685. https://doi.org/10.18653/v1/2022.acl-long.529.

Haruechaiyasak, C., Kongyoung, S., & Dailey, M. (2008). A comparative study on Thai word segmentation approaches. *2008 5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 1*, 125–128. https://doi.org/10.1109/ECTICON.2008.4600388.

Kittinaradorn, R., Chaovavanich, K., Achakulvisut, T., Srithaworn, K., Chormai, P., Kaewkasi, C., Ruangrong, T., & Oparad, K. (2019). *DeepCut: A Thai Word Tokenization Library using Deep Neural Network*. (v1.0). Zenodo. https://doi.org/10.5281/zenodo.3457707.

National Electronics and Computer Technology Center. (2022). LST20 Corpus. https://opend-portal.nectec.or.th/it/dataset/lst20-corpus. (In Thai)

Noyunsan, C., Haruechaiyasak, C., Poltree, S., & Saikaew, K.R. (2014). A Multi-Aspect Comparison and Evaluation on Thai Word Segmentation Programs. *Joint International Conference of Semantic Technology*. https://ceur-ws.org/Vol-1312/jist2014pd_paper6.pdf.

Poowarawan, Y. (1986). Dictionary-based Thai Syllable Separation. *In Proceedings of the Ninth Electronics Engineering Conference*, 409–418.

Ruenlek, K., & Damrongkamoltip, K. (2023). *A Mockup Dataset for Word Segmentation Based on the LST20 Dataset*. https://github.com/Fairpart/A-mockup-dataset-for-word-segmentation-based-on-the-LST20-DATASET.

Theeramunkong, T., Sornlertlamvanich, V., Tanhermhong, T., & Chinnan, W. (2000). Character Cluster based Thai Information Retrieval. *Proceedings of the Fifth International Workshop on on Information Retrieval with Asian Languages*, 75–80. https://doi.org/10.1145/355214.355225.

Virach, S. (1993). Word Segmentation for Thai in Machine Translation System. *Machine Translation*. 50-56. (In Thai)

Citation:
Damrongkamoltip, K., Ruenlek, K., Limprasert, W., & Boonkwan, P. (2024). Comparing a Thai Words Segmentation Methods in the LST20 Dataset. *Journal of Computer and Creative Technology, 2*(2), 61-70. https://doi.org/10.14456/jcct.2024.7.