Natural Language Processing for Subjective Exam Grading in Programming Explanations in Thai Language

Waris Rattanimit*
Electrical Technology and Automation System,
Faculty of Engineering and Technology
Siam Technology College
Bangkok, Thailand
warisr@siamtechno.ac.th

Thanyarat Nomponkrang
Department of Computer Education
King Mongkut's University of Technology North Bangkok
Bangkok, Thailand
thanyaratn@kmutnb.ac.th

2025 13th International Electrical Engineering Congress (iEECON) | 979-8-3315-4395-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/iEECON64081.2025.10987594

***Abstract*—In the digital era, learning programming has become an essential skill, and assessing students' understanding in this area plays a crucial role in enhancing educational quality. However, evaluating subjective programming exam responses, especially those involving conceptual explanations or procedural descriptions, remains a challenging task due to the diversity and complexity of answers. Manual grading can lead to delays and inconsistencies in scoring. This research aims to develop an automated system for evaluating Thai subjective programming exams using Natural Language Processing (NLP) techniques to assess the correctness of responses. The system is implemented in Python, utilizing the ThaiNLP library for word tokenization, alongside Machine Learning algorithms such as Naive Bayes, Decision Tree, and Support Vector Machine (SVM) for classification and prediction of answers. The experimental dataset includes responses from 120 students who answered subjective questions about Python commands. Expert testing indicated high user satisfaction with an average score of 4.05. The findings show that SVM achieved the highest accuracy at 85.33%. This system can enhance efficiency and consistency in exam evaluation, reducing grading time and contributing to standardized scoring.**

Keywords— Programming learning, Subjective exam evaluation, Natural Language Processing (NLP), Word tokenization, Machine Learning

I. INTRODUCTION

In the digital age where technology plays a pivotal role in education, learning to code has become an essential skill for professionals across various fields. Evaluating students' programming comprehension is crucial for improving the quality of education. However, grading subjective questions related to explaining programming concepts or processes remains challenging due to the diversity and complexity of student responses. Relying solely on human graders can lead to delays and inconsistencies in scoring.

II. LITERATURE REVIEW

A. Machine Learning

Machine Learning as the process of using data to enable computers to learn independently. This can be categorized into three primary types [7].

- Supervised Learning: Supervised learning involves training a computer using labeled data, where both the input and the expected output are provided. The algorithm learns to map inputs to outputs based on the labeled examples.It requires labeled datasets and aims to minimize the difference between predicted and actual outputs, thereby reducing error. This type of learning is divided into two subcategories: classification, which deals with discrete outputs and regression, which deals with continuous outputs. Applications include predictive maintenance in industrial settings, fraud detection in financial transactions, and medical diagnosis from patient data.
- Unsupervised Learning: In unsupervised learning, the algorithm discovers patterns in unlabeled data without being explicitly told what to look for. It identifies inherent structures in the data.This type of learning works with unlabeled datasets and focuses on exploring the data structure rather than prediction. Common techniques include clustering, which groups similar data points together and dimensionality reduction, which simplifies data for visualization or preprocessing. Applications include market segmentation to understand customer behavior, image compression and feature extraction, and preprocessing for supervised learning tasks.
- Reinforcement Learning: Reinforcement learning involves training an agent to make decisions by interacting with an environment. The agent learns through trial and error, receiving rewards or penalties based on its actions.This type of learning focuses on decision-making in dynamic environments and is driven by feedback. The core elements include the agent, the environment, and the reward signal. Applications include developing AI agents for gaming, autonomous robotics for navigation and task execution, and optimizing dialogue systems in natural language processing.

This research aims to develop a Thai language essay grading system based on similarity measures, inspired by the work of Rakchatphum Wongsarak, Khanitha Homchan, and Nongnuch Ketuey (2024). Their study successfully developed a Thai essay grading system using similarity measures. The system is designed for both instructors and students. Building upon their foundational research, this study focuses on extending the capabilities of the system by developing a web application that incorporates advanced Natural Language Processing (NLP) techniques. The application is designed to be a practical tool for both instructors and students. For instructors, the system offers features for automated grading, detailed feedback, and analytics to identify common areas where students struggle. For students, it provides instant feedback on essay quality, enabling them to identify

*Corresponding author
979-8-3315-4395-2/25/$31.00 @2025 IEEE

weaknesses and improve their writing skills iteratively.The implementation leverages Python programming language and state-of-the-art NLP libraries to ensure accurate and efficient processing of Thai text. Techniques such as tokenization, semantic similarity calculations, and syntactic analysis are employed to assess essays comprehensively. demonstrating the system's potential as a valuable tool in the educational context. Evaluation by experts has shown that the system is effective and user-friendly, with a high average satisfaction rating of 4.05 [3].

III. METHODOLOGY

A. Data Collection and Preparation

A dataset was compiled by administering a programming assessment through Google Forms. The assessment was designed for students enrolled in a programming course to evaluate their understanding of Python commands. A total of 120 students participated, each tasked with providing written explanations for a specific Python command. This method aimed to capture their conceptual understanding and ability to articulate the functionality and purpose of Python commands in their own words.

The collected responses offer insights into students' comprehension levels, common misconceptions, and the effectiveness of instructional strategies in teaching Python programming. This dataset could serve as a basis for further analysis,and exploring correlations between their explanations and overall learning outcomes.

B. Machine Learning Implementation

The Altair AI Studio (RapidMiner) platform was leveraged to develop and train machine learning models for automated essay grading. These models were tasked with evaluating the quality and relevance of student responses.

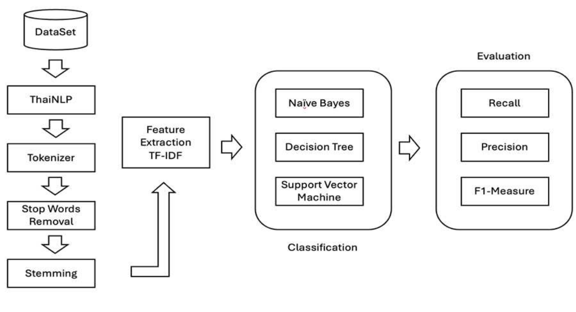

Fig. 1. System Methodology

Dataset consisted of subjective responses to a single, open-ended Python programming question. Participants were tasked with providing a written explanation of a given Python code snippet. The subjective test responses collected through Google Forms were exported as a CSV file from Google Sheets.

	A	B
1	question	score
2	การใส่จำนวนแล้วคูณกับ2.4	Y
3	ใส่เลขในinput คำตอบที่ออกมาคือเลขที่ใส่คูณ2.4	Y
4	การใส่จำนวนแล้วคูณกับ2.4	Y
5	การใส่จำนวนแล้วคูณกับ2.4	Y
6	คูณค่าตัวเลขที่ใส่เข้ามาแล้วแปลงเป็นหน่วย CM	Y
7	ไว้ในคำนวน หรือ แปรงค่าตัวเลขที่มีจุดทศนิยม	Y
8	เขียนให้รันตลอดเวลา	N
9	จะทำให้ตัวแปร h เก็บจำนวนจริง 2.4	Y
10	สร้าง Loop ที่ไม่มีที่สิ้นสุด	N

Fig. 2. Labeled Dataset

The CSV file containing subjective responses was annotated with binary labels ('Y' for correct, 'N' for incorrect) to generate an exampleset for machine learning shown in Fig2.

Next, we used a Python library called PyThaiNLP to break down the text into individual words. This process, known as word tokenization, is a common step in natural language processing. The results of this step are shown in the Fig3.

```
การ|ใส่|จำนวน|แล้ว|คูณ|กับ|2.4|,Y
ใส่|เลข|ใน|input| |คำตอบ|ที่|ออกมา|คือ|เลขที่|ใส่|คูณ|2.4|,Y
การ|ใส่|จำนวน|แล้ว|คูณ|กับ|2.4|,Y
การ|ใส่|จำนวน|แล้ว|คูณ|กับ|2.4|,Y
คูณ|ค่าตัว|เลขที่|ใส่|เข้ามา|แล้ว|แปลง|เป็น|หน่วย| |CM|,Y
ไว้|ใน|คำ|นว|น| |หรือ| |แปรง|ค่าตัว|เลขที่|มี|จุดทศนิยม|,Y
เขียน|ให้|รัน|ตลอดเวลา|,N
จะ|ทำให้|ตัวแปร| |h| |เก็บ|จำนวนจริง| |2.4|,Y
สร้าง| |Loop| |ที่|ไม่|มี|ที่|สิ้นสุด|,N
สร้าง| |Loop| |ที่|ไม่|มี|ที่|สิ้นสุด|,Y
สร้าง| |Loop| |ที่|ไม่|มี|ที่|สิ้นสุด|,Y
สร้าง| |Loop| |ที่|ไม่|มี|ที่|สิ้นสุด|,Y
ทำให้|คำสั่ง|รัน|ตลอดเวลา|,Y
```

Fig. 3. Break the Text Down

To preprocess the data using Altair AI Studio or RapidMiner, follow these steps shown in Fig.4.

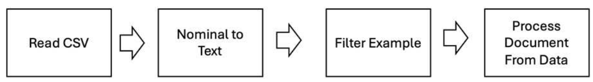

Fig. 4. Operator Preparation with Altair Ai Studio

Data Preparation

Data Input: Import the CSV file containing ThaiNLP-tokenized text.

Data Cleaning: Convert the relevant column to a text data type and remove records with missing values.

Feature Engineering

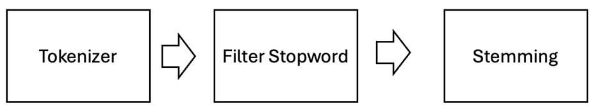

Fig. 5. Process Document Form Data

Tokenization: Apply ThaiNLP-based tokenization to split text into individual tokens. Word tokenization is the process of splitting text into individual words or tokens. It's the step in many NLP tasks.

Filter Stop Word: Utilize a stop word list (e.g., stopword-th.txt) to eliminate common, less informative words.

Stemming: Employ a stemming algorithm to reduce words to their root form.

TF-IDF: Calculate Term Frequency-Inverse Document Frequency to quantify the importance of terms within the documents and the corpus. All Step Process shown in Fig.5.

In this experiment, we employed three algorithms to compare and identify the optimal model for prediction shown in Fig.1.

Naive Bayes is a fundamental probabilistic machine learning algorithm primarily utilized for classification tasks. Rooted in Bayes' Theorem, it presumes that all features are independent of one another when conditioned on the class label. While this assumption of feature independence may not always reflect real-world relationships, it significantly simplifies the computational process, enabling the algorithm to operate with remarkable efficiency. Its straightforward nature and ability to handle large datasets make it a popular choice for a variety of applications, particularly in scenarios where quick and reliable results are needed.

Decision Tree is a supervised learning algorithm used for both classification and regression tasks. It constructs a tree-like model of decisions by recursively partitioning the data based on specific attributes. Each internal node represents a decision on an attribute, each branch represents the outcome of the decision, and each leaf node represents a class label or a continuous value for regression. This algorithm is widely chosen for its interpretability, as it provides a visual representation of decision-making processes, and its ability to handle both categorical and continuous data effectively.

Support Vector Machine (SVM) is a powerful supervised learning algorithm primarily used for classification tasks. It aims to find the optimal hyperplane that best separates data into two classes. This hyperplane maximizes the margin between the classes, ensuring robustness to classification errors. SVM is particularly effective for high-dimensional spaces and is well-suited for applications where the number of dimensions exceeds the number of samples. It can also handle non-linear relationships using kernel functions.

The multiply operator was used to duplicate the ExampleSet, and the Cross Validation operator was utilized to incorporate the three machine learning algorithms. SVM is particularly suitable for binary classification problems with clear margin separation between classes.

Within each fold of cross-validation, the algorithm is trained on a specific training set. Subsequently, the trained model is applied to a corresponding testing set to evaluate its performance.

After running the program, we obtain a word list generated from the TF-IDF process. This results in a data example set, which is essentially a data table containing tokenized words, their corresponding TF-IDF weights, and the associated labels (correct or incorrect) for prediction purposes.

Evaluation: Evaluating algorithm performance is crucial to ensure the system's accuracy and reliability. The following metrics are used

- Recall: The model's ability to find all relevant instances. In the context of essay grading, it reflects the system's capacity to detect and evaluate all key elements within the text that contribute to its quality and relevance.
- Precision: The accuracy of positive predictions. For essay grading, this means that the feedback provided by the system accurately identifies the strengths and weaknesses of the essay without misattribution. High precision ensures that students and instructors receive meaningful and relevant feedback.
- F1-score: A combined measure of precision and recall. It is especially useful when there is a need to find a trade-off between identifying all relevant elements (recall) and maintaining accuracy in predictions (precision). In this system, a high F1-score indicates that the grading mechanism is both comprehensive and accurate, making it reliable for practical use.

TABLE I. COMPARISON PERFORMANCE THREE ALGORITHM

Model	Accuracy	Precision (Y)	Precision (N)	Recall (Y)	Recall (N)
Decision Tree	83.00% ± 13.74%	80.77%	85.71%	84.00 %	82.76 %
Support Vector Machine	85.33% ± 11.13%	100.00%	78.38%	68.00 %	100.00%
Navie Bayes	84.67% ± 15.65%	84.00%	86.21%	84.00 %	86.21 %

The Support Vector Machine (SVM) algorithm exhibited the highest accuracy of 85.33% and was therefore selected for the prediction task.

To evaluate the model's performance on unseen data, two unlabeled data points were processed using the same preprocessing steps as the training data.


Text Sentiment
ใส่จำนวนแล้วคูณกับ2.4ได้ออกมาเป็นเซ็นติเมตร
วนลูปรับค่า

Fig. 6. Unseen Data

Significantly, the word list generated from the initial Process Documents form Data was utilized as a reference in a subsequent Process Documents form Unseen Data step.

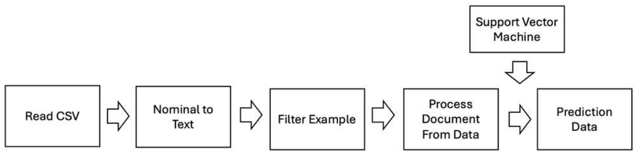

Fig. 7. Process Document Unseen Data

The trained SVM model was then applied to these new data points. The prediction results indicated that the first data point had a high probability of being classified as positive (Y), with a confidence score of 0.76

TABLE II. COMPARISON PERFORMANCE THREE ALGORITHM

	Prediction	Confidence(Y)	Confidence(N)
1	Y	0.760	0.240
2	N	0.325	0.675

The probability of being classified as negative (N) was 0.24. Conversely, the second data point was more likely to be classified as negative (N), with a confidence score of 0.675 for the negative class and 0.325 for the positive class.

IV. DISSCUSSION

This research result into the application of natural language processing techniques to automate the assessment of programming explanations. By adopting established NLP methodologies such as tokenization, as demonstrated in the work of Thachpichya Chamnankit and Thitirat Siribunwattanakul on Thai fake news detection [1], and building upon sentiment analysis techniques explored by Phisit Bovoralesisuthee and Worapat Pairikerng for classifying online product reviews, this study aimed to develop a robust model for code explanation assessment [2]. Following the paradigm set by Suda Tipyaprasert in developing a depression risk assessment model using social media data, this research employed a machine learning approach to classify code explanations [8].

A comparative analysis of three algorithms Naive Bayes, Decision Tree, and Support Vector Machine was conducted. The results indicated that the Support Vector Machine achieved the highest accuracy of 85.33%, making it the most suitable algorithm for the task. Subsequent prediction experiments on unseen data further validated the model's effectiveness.

The implications of this study are significant. First, it establishes a foundation for automating assessments in programming education, reducing reliance on manual grading and mitigating subjective biases.

Educators can leverage this system to provide instant feedback, thereby enhancing the learning experience. Second, the methodology can be extended to other domains requiring the evaluation of textual explanations, such as technical documentation, scientific writing, and online learning platforms

While this research achieved promising results, there are opportunities for further refinement and expansion. For instance, incorporating deep learning models like Transformer-based architectures could potentially enhance the model's understanding of context and semantic nuances in programming explanations. Additionally, integrating domain-specific features, such as the complexity of code snippets and the alignment of explanations with programming best practices, could improve the assessment's accuracy.

V. SUMMARIZE

This research explores the potential of natural language processing (NLP) techniques in automating the grading of subjective programming assignments. The developed system demonstrates the ability to efficiently and accurately evaluate student responses, reducing the workload of instructors and minimizing errors that may arise from human fatigue or inconsistencies in grading. By systematically analyzing student answers, even those with diverse expressions, the system can accurately assess responses through similarity measures and classification tasks. The results of this research highlight the feasibility of utilizing NLP technology to enhance the quality of education and assessment. However, improvements are still needed in terms of the system's scalability and its capacity to process large datasets.

REFERENCES

[1] T. Chamnankit and T. Siribunwattanakul, "Detection of Thai Fake news Using Natural Language Processing Techniques," Kasetsart Journal of Social Sciences, vol. 45, no. 2, pp. 275-287, 2022. doi:10.14456/kmuttrd.2022.16.

[2] P. Bovoralesisuthee and W. Pairikerng, "Sentiment Analysis Model for classifying online product review articles," Journal of Engineering and Digital Technology (JEDT), vol. 10, no. 1, pp. 71-79, 2022.

[3] R. Wongsara, K. Homchan, and N. Ketuey, "Development of a Thai Subjective Question Answering System Based on Similarity measurement," Journal of Applied Information Technology, vol. 17, no. 1, pp. 11-21, 2021.

[4] N. Pleerux and A. Nardkulpat, "Applying topic modeling and sentiment analysis to understand customer online reviews of restaurants during the COVID-19 pandemic in Thailand," Heliyon, vol. 9, no. 11, p. e22193, 2023. doi:10.1016/j.heliyon.2023.e22193.

[5] R. R. J. Jardim, C. Delgado, and M. F. Silva, "CLIQ_v2.2.py: An open-source software for classifying questions into cognitive levels," Software Impacts, vol. 13, p. 100345, 2022. doi:10.1016/j.simpa.2022.100345.

[6] I. Goenaga, A. Atutxa, K. Gojenola, M. Oronoz, and R. Agerri, "Explanatory argument extraction of correct answers in resident medical exams," Artificial Intelligence in Medicine, vol. 157, p. 102985, 2024. doi:10.1016/j.artmed.2023.102985.

[7] W. Rattanamit, "Development of a Training Set on The Internet of Things and Artificial Intelligence, Especially Machine Learning, for Upper Secondary School Students in Bangkok Noi district," in Proc. 14th Benchamitra Academic National and International Conf., Chiang Mai University, Chiang Mai, Thailand, 2024, pp. 1-11.

[8] S. Tipyaprasert, "Development of a Depression Risk Assessment Model Using Social Media Data," Srinakharinwirot University Science and Technology Journal, vol. 16, no. 31, pp. 1-16, 2023.

[9] P. Choksuwatansakul, S. Artsaiboon, and A. Tarongrattanarit, "Application of Data Dcience in Analyzing Data from Impact Assessment and Evaluation of the Law," Journal of Law, Thammasat University, vol. 52, no. 2, pp. 247-273, 2022.

[10] B. Hantedsitth, "Analysis of medical students' opinions on forensic medicine using artificial intelligence," Journal of Maharat Nakhon Ratchasima Hospital, vol. 46, no. 2, pp. 262-273, 2022. doi:10.33165/rmj.2023.46.2.2624