IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 12 January 2024, accepted 17 February 2024, date of publication 20 February 2024, date of current version 27 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3367994

RESEARCH ARTICLE

Prompt-Based Label-Aware Framework for Few-Shot Multi-Label Text Classification

THANAKORN THAMINKAEW[1], PIYAWAT LERTVITTAYAKUMJORN[2], AND PEERAPON VATEEKUL[1], (Member, IEEE)

[1]Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Pathum Wan, Bangkok 10330, Thailand
[2]Google LLC, Mountain View, CA 94043, USA

Corresponding author: Peerapon Vateekul (peerapon.v@chula.ac.th)

ABSTRACT Prompt-based learning has demonstrated remarkable success in few-shot text classification, outperforming the traditional fine-tuning approach. This method transforms a text input into a masked language modeling prompt using a template, queries a fine-tuned language model to fill in the mask, and then uses a verbalizer to map the model's output to a predicted class. Previous prompt-based text classification approaches were primarily designed for multi-class classification, taking advantage of the fact that the classes are mutually exclusive and one example belongs to only one class. However, these assumptions do not hold in the context of multi-label text classification, where labels often exhibit correlations with each other. Therefore, we propose a Prompt-based Label-Aware framework for Multi-Label text classification (PLAML) that addresses the challenges. Specifically, PLAML enhances prompt-based learning with three proposed techniques to improve the overall performance for multi-label classification. The techniques include (i) a token weighting algorithm that considers the correlations between labels, (ii) a template for augmenting training samples, making the training process label-aware, and (iii) a dynamic threshold mechanism, refining the prediction condition of each label. Extensive experiments on few-shot text classification across multiple datasets with various languages show that our PLAML outperforms other baseline methods. We also analyzed the effect of each proposed technique to better understand how it is suitable for the multi-label setting.

INDEX TERMS Few-shot learning, multi-label classification, natural language processing, prompt-based learning, text classification, verbalizer.

I. INTRODUCTION

Prompt-based learning has received considerable attention in the field of natural language processing (NLP) in recent years. So far, a few research papers have presented compelling results of its effectiveness in multi-label text classification (MLTC) [1], [2]. This approach fundamentally seeks to replicate the training process of a language model (LM) using masked language modeling, which serves to bridge the gap between the pre-training task and the downstream task. It diverges from traditional methods that involve training or fine-tuning with additional parameters when tuning extra classifiers.

A common approach to implementing prompt-based learning involves wrapping a text input with a predefined template and using a pre-trained or fine-tuned LM to do a masked language modeling task. For instance, consider a text input "I like your optimism!" with two distinct labels: "joy" and "optimism". We can prompt a masked LM with "I like your optimism! It was `[MASK]`." Then, the model may predict "peace" for the `[MASK]` token. Next, we use a function known as a "verbalizer" to map "peace" back to the original labels "joy" and "optimism".

Extended research in verbalizers for text classification tasks has been conducted, categorizing them into three main approaches: a manual verbalizer, which relies on label names; a discrete verbalizer, which employs algorithms to identify representative words of each label; and a continuous verbalizer, which uses algorithms to identify a representative

The associate editor coordinating the review of this manuscript and approving it for publication was Wei-Yen Hsu.

vector of each label [3]. However, in the context of MLTC, most works have focused primarily on the manual verbalizer [1], [2]. Nonetheless, implementing the manual verbalizer can pose challenges, given that label names may not be optimal with respect to the underlying LM. Moreover, certain label names may not be accurately summarized by a single token. These challenges become particularly evident when applying prompt-based learning to a dataset without having an expert to select and verify the quality of the manually chosen representative words [4], [5], [6].

To address these challenges, a discrete verbalizer is gaining preference, as it automatically searches representative tokens while still providing interpretable outputs similar to a manual verbalizer. Recent methods in discrete verbalizers aim to leverage the knowledge of label names, resulting in better performances compared to a manual verbalizer approach. This can be accomplished by incorporating external knowledge into discrete verbalize [7], performing an embedding similarity to the manual label name [5], or including label names to a verbalizer template [8]. However, these discrete verbalizers had been primarily experimented with in the context of multi-class classification. Applying them to MLTC may not be straightforward due to their assumption that each token in the vocabulary can only be assigned to at most one label [4], [8]. This assumption is often considered unrealistic, as labels in MLTC are frequently closely related, resulting in some labels lacking good representative tokens.

In this paper, we propose a novel Prompt-based Label-Aware framework for Multi-Label text classification (PLAML), a framework that aims to enhance the overall effectiveness of the prompt-based learning approach for MLTC in few-shot settings. The main contributions can be summarized as follows:

- To the best of our knowledge, this paper is the first attempt to apply a prompt-based learning approach using discrete verbalizers to MLTC.
- We propose PLAML that consists of three components – an improved weighting approach for each representative token based on its label frequency, an upgraded label-aware template to augment training samples, and a dynamic threshold mechanism assigning each label with different prediction thresholds.
- We conduct few-shot classification experiments on three datasets from three languages. The results show that PLAML outperforms the baselines in most settings.
- We carry out additional analyses to determine and improve each of the three components of the proposed framework.

The rest of this paper is organized as follows. Section II provides related work on three relevant topics of PLAML–multi-label text classification, prompt-based learning, and verbalizers. Section III explains our PLAML framework, including the problem statement, the framework overview, and the detailed steps of the framework. Section IV describes the settings of our experiments. After that, Section V discusses the results of the whole framework and the effect of each component of PLAML. Finally, the paper is concluded in Section VI.

II. RELATED WORK

A. MULTI-LABEL TEXT CLASSIFICATION (MLTC)

MLTC is a task of predicting one or more labels for each text input. Traditional text classification approaches use embeddings derived from models such as Word2Vec [9] or contextual embeddings from pre-trained language models (PLMs) like Bidirectional Encoder Representations from Transformers (BERT) [10] and Robustly optimized BERT approach (RoBERTa) [11]. Subsequently, these embeddings are fed into a neural network layer, which may comprise a linear layer or recurrent components such as Long Short-Term Memory (LSTM) [12], or convolutional elements like convolutional neural network (CNN) [13]. In the final neural network layer, a sigmoid activation function is frequently employed to generate probability outputs for each label. While this approach effectively captures semantic relationships between words, it overlooks label dependencies and correlations in the multi-label setting. Moreover, additional parameters are introduced when tuning the classifiers on top of the `[CLS]` which imposes the necessity for a larger training dataset to enhance the model's performance [14].

More effective approaches involve considering that a label can be associated with multiple other labels. Yang et al. [15] proposed a sequence generation model (SGM) that treats MLTC tasks as a sequence generation problem and uses a recurrent neural network (RNN) decoder structure to capture the dependencies among labels. However, sequential models require an extensive search for the optimal solution within the potential label space, a process that can become time-intensive when dealing with a large number of labels. Meanwhile, Pal et al. [16] proposed a method called Multi-label Text Classification using Attention-based Graph Neural NETwork (MAGNET), which embraces Graph Neural Networks (GNNs) [17] to capture dependencies among labels using a feature matrix and a correlation matrix before generating classifiers for the task. On a different front, Ananiadou [18] proposed a model for casting multi-label emotion classification as span-prediction (SpanEmo) that views MLTC as a span prediction task. It combines text input and labels within a single input for selecting a span of output labels, to help the PLM learn associations between labels in a given sentence. The method also introduces a loss function specifically created to model the presence of multiple co-existing labels within the input sentence. While these approaches proved to be effective in full dataset settings, they have not been tested in few-shot settings.

B. PROMPT-BASED LEARNING

Prompt-based learning has emerged as a preferred approach when there is a shortage of training data. This is because it leverages existing tokens that the PLM is already familiar with, as opposed to the `[CLS]` token, which would require

the PLM to tune both [CLS] token and neural network layers from scratch.

The prompt-based approach consists of three critical elements: a template, a verbalizer, and a [MASK] token [3]. A template refers to a phrase or sentence that guides the LM to understand the contextual intention of the prediction. Additionally, it transforms the text input into a format suitable for the PLM to generate predictions. A verbalizer is pivotal in converting the output from the LM into human-readable labels. Meanwhile, a [MASK] token serves as a dynamic placeholder for the input text, allowing the LM to contextually analyze and generate predictions for multiple labels simultaneously.

Several papers have utilized prompt-based learning in the context of MLTC. Song et al. [19] proposed Label Prompt Multi-label Text Classification (LP-MTC), utilizing a prefix template with special tokens to capture label associations through self-attention mechanisms. Wang et al. [2] adopted a prompt-based methodology for medical text classification in Chinese, employing discriminative PLMs—ELECTRA Efficiently Learning an Encoder that Classifies Token Replacements Accurately) [20] and (ERNIE-Health (Enhanced Representation from kNowledge IntEgration in Health) [21]. Additionally, the authors used [UNK] instead of [MASK] to compel the model discriminator to make predictions from candidate words. Similarly, Wei et al. [1] proposed Prompt Tuning for Multi-Label Text Classification (PTMLTC), a prompt-based strategy using the label name itself as a verbalizer to establish connections between textual inputs and knowledge concepts. While the former two studies conducted experiments on full datasets, the latter primarily focused on few-shot scenarios. However, these approaches did not emphasize improving a verbalizer, which has a strong influence on the performance of prompt-based learning [7], [22].

C. VERBALIZERS FOR PROMPT-BASED LEARNING

Manually selecting a single token to represent each label, as in PTMLTC [1] and Pattern Exploiting Training (PET) [14], is the simplest way to construct a verbalizer. However, manual selection could be laborious, given that datasets in MLTC usually have many labels, and it does not guarantee the optimal selection of tokens when conditioned on the chosen LM.

To automate verbalizer construction, Hambardzumyan et al. [23] introduced trainable continuous tokens to serve as label representations, known as a continuous verbalizer. Nonetheless, the acquired tokens may not correspond to actual words, making it harder to debug and improve the model. Ji et al. [24] proposed Hierarchical Verbalizer (HierVerb) that builds a verbalizer on top of continuous verbalizers designed specifically for a hierarchical text classification problem. Meanwhile, other works favor discrete verbalizers because of their enhanced interpretability. Ji et al. [6] searched for the best token to represent each label by maximizing the likelihood of the training data, referred to as PET with Automatic Labels (PETAL). Schick et al. [4] introduced Automatic Multi-Label Prompting (AMuLaP), which does the same but represents each class by multiple tokens instead to reduce the effects of noise in the data. However, most automatically generated verbalizers do not perform as well as manually selected verbalizers [4], [7], [25].

As a result, recent approaches to discrete verbalizers aim to leverage the knowledge of label names, resulting in improved effectiveness compared to the manual method. Hu et al. [26] proposed Knowledgeable Prompt Tuning (KPT), which incorporates external knowledge bases into discrete verbalizers. However, acquiring external knowledge for low-resource language datasets has proven to be a formidable challenge. Zhao et al. [5] proposed NonParametric Prompting (NPPrompt), which represents each class using a set of tokens with the highest embedding similarity to the manual label. Its performance, therefore, heavily relies on the quality of the LM's embedding space, which may not be efficacious for mid-to-low resource languages.

In our previous work, Label-Aware Automatic Verbalizer (LAAV), we used class labels along with a conjunction word, ''and'', to help the PLM generate more relevant words for the discrete verbalizer. This leads to a better selection of words compared to using the label name or other automatic algorithms approaches as in AMuLaP [4] that only use the text inputs. Furthermore, LAAV was tested on multiple low-resource language datasets and demonstrated greater effectiveness compared to other discrete verbalization approaches. Nonetheless, it is worth noting that these discrete verbalizers including LAAV were originally experimented with the multi-class classification problem [4], [5], [6], [8], [26]. Therefore, our experiments begin with implementing the current discrete verbalizers to MLTC by allowing them to select the same representative token for multiple labels. Next, we propose PLAML, a new framework that aims to enhance the overall effectiveness of the prompt-based learning approach for MLTC in few-shot settings.

III. METHODOLOGY

In this section, we begin with a brief introduction of the problem statement in Section III-A followed by the overview of our PLAML framework for few-shot MLTC in Section III-B. After that, we detail each step of PLAML in Sections III-C to III-G.

A. PROBLEM STATEMENT

The goal of multi-label text classification is to predict the labels to which a given input text belongs. Formally, let $\mathcal{L} = \{l_1, l_2, \ldots, l_L\}$ be a set of all possible L labels in this classification task, and $\mathcal{D}_{train} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a training dataset where x_i is the i^{th} input text and $y_i \in \{0, 1\}^L$ represents the labels of x_i. Specifically, $y_{ij} = 1$ if x_i has the label l_j; otherwise, 0. Importantly, as each input text can be associated with multiple labels, this results in multiple $y_{ij} = 1$ values.

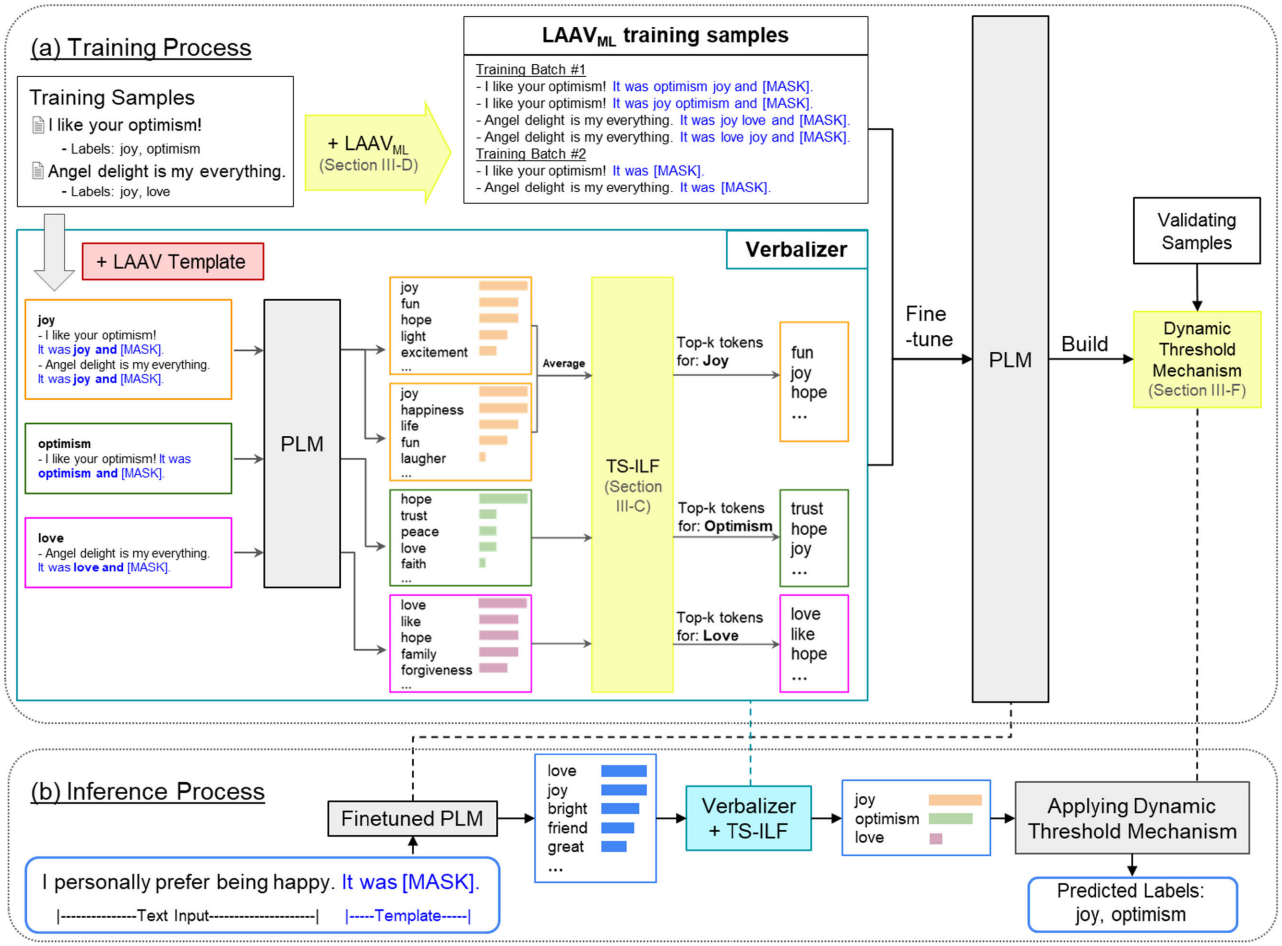

FIGURE 1. PLAML: a prompt-based learning framework for MLTC comprising three proposed techniques including TS-ILF, *LAAV*$_{ML}$, and dynamic threshold mechanism. Solid lines represent operations within either the training or the inference process, while dashed lines connect the corresponding components between the training and the inference processes.

We aim to train a machine-learning framework that can output the predicted labels $\hat{y} \in \{0, 1\}^L$ for an unseen input text x. In this paper, we also assume the availability of the validation dataset $\mathcal{D}_{val}$ with the same format as $\mathcal{D}_{train}$.

B. FRAMEWORK OVERVIEW

PLAML applies prompt-based learning using an automatic discrete verbalizer (similar to [4], [5], [26]) for few-shot MLTC. Figure 1 shows the entirety of PLAML, with highlighted areas in yellow representing the major contributions presented in this paper. The training process of PLAML can be summarized in four steps.

The first step is **verbalizer construction**, illustrated in the light-blue area of Figure 1, where we find a set of tokens that represent each label and assign an appropriate weight to each token. In particular, for each label l_i, this step returns $\mathcal{S}(l_i) = \{(v_{i,1}, w_{i,1}), \ldots, (v_{i,k}, w_{i,k})\}$ where $w_{i,j}$ is the weight of the token $v_{i,j}$ while k is the number of tokens representing each label.

The second step is **training sample generation**, applying our newly proposed templates to $\mathcal{D}_{train}$ to generate actual samples for prompting. This process is referred to in the top part of Figure 1 (a) between the training samples and $LAAV_{ML}$ training samples.

The third step is **fine-tuning the PLM** using the verbalizer from the first step and the generated samples from the second step. The LM used in this paper are the masked LM.

Finally, the fourth step is **setting label-specific thresholds** using the fine-tuned PLM and the validation dataset $\mathcal{D}_{val}$, as shown on the right-hand side of Figure 1. These thresholds will be used together with the verbalizer to map the output of the fine-tuned PLM into predicted labels during inference.

The following subsections describe each of the training steps and the inference step in detail, while our novel contributions lie mainly in the verbalizer construction, the training sample generation, and the threshold mechanism steps of the training process.

C. VERBALIZER CONSTRUCTION

First, we extend the idea of LAAV [8], as shown in the light-blue area of Figure 1, to find k representative tokens of each label. To recap, for each example x with the label

l_i, LAAV uses the label-aware template

$$T_{l_i}(x) = [x]\ It\ was\ [l_i]\ and\ \texttt{[MASK]} \tag{1}$$

to query the probability score of every possible token v from the PLM. Considering the whole $\mathcal{D}_{train}$, the score of token v for the label l_i is

$$s(v, l_i) = \sum_{x \in \mathcal{X}_i} p_M(\texttt{[MASK]} = v | T_{l_i}(x)) \tag{2}$$

where $\mathcal{X}_i$ is the set of all examples in $\mathcal{D}_{train}$ that have the label l_i and p_M is the probability score given by the PLM. LAAV ensures that each token v can be assigned to only one label. So, it assigns v to the label l_i where $l_i = \arg\max_{l \in \mathcal{L}} s(v, l)$. After obtaining the k best tokens for each label, it gives each of them an equal weight during training and inference.

While LAAV works well for multi-class text classification [8], the fact that it assigns one token to at most one class incurs impracticality in MLTC. This is because labels in MLTC are often closely related, leading to some tokens achieving high scores for more than one label. Assigning such tokens to a single label may make some labels lose important representative tokens or even have no informative representative tokens. However, allowing one token to associate with multiple labels is problematic if we give every token an equal weight as in LAAV because the selected tokens may have different levels of importance for each label. For example, a token v may be related to both labels l_1 and l_2 but it is more important for l_1 than l_2. Hence, PLAML chooses to allow a token to represent more than one label and assign an appropriate weight for each token instead as explained in the framework overview.

To explain, PLAML calculates $s(v, l_i)$ for every label l_i and token v in the vocabulary. Then it keeps only the top k tokens with the highest scores for each label. Let $\mathcal{V}(l_i)$ denote the set of representative tokens of l_i. One obvious way for weight assignment is using $s(v, l_i)$ as the weight of the token v for the label l_i. Nonetheless, this might not yield optimal results as tokens with such high scores can be associated with multiple labels, making them less useful for classification. Inspired by the information retrieval concept, namely Term Frequency-Inverse Document Frequency (TF-IDF) [27], PLAML introduces a new weighting approach that takes into consideration both the probability score associated with each token and its occurrence frequency across the set of labels. Our proposed method, Token Score - Inverse Label Frequency (TS-ILF), can be explained in two parts.

Token score (TS) of v for l_i is the $s(v, l_i)$ normalized so that the TS of all representative tokens of l_i sums up to 1. Mathematically,

$$TS(v, l_i) = \frac{s(v, l_i)}{\sum_{v' \in \mathcal{V}(l_i)} s(v', l_i)} \tag{3}$$

Inverse Label Frequency (ILF) is determined by first identifying the frequency of each token across the set of labels and then applying the inverse document frequency equation.

$$ILF(v) = log \frac{L}{|\{l_i | v \in \mathcal{V}(l_i)\}|} \tag{4}$$

where L is the number of labels and $|\{l_i | v \in \mathcal{V}(l_i)\}|$ is the number of labels of which v is one of the representative tokens.

Now, the weight of token v for the label l_i is defined as

$$w(v, l_i) = TS\text{-}ILF(v, l_i) = TS(v, l_i) \times ILF(v) \tag{5}$$

Finally, the TS-ILF will be applied to calculate the score of the label l_i for a text input x_j as follows:

$$S(l_i | x_j) = \frac{1}{k} \sum_{v \in \mathcal{V}(l_i)} w(v, l_i) p_M(\texttt{[MASK]} = v | T(x_j)) \tag{6}$$

Then $S(l_i|x_j)$ could be converted into the probability of label l_i using the sigmoid function.

$$p_{ji} = p(y_{ji} = 1 | x_j) = \text{sigmoid}(S(l_i | x_j)) \tag{7}$$

D. TRAINING SAMPLE GENERATION

LAAV and other previous studies [4], [6], [14] used the same prompt template during training and testing to prevent the out-of-distribution problem, which is indicated in the top section of Figure 1 marked "Training Batch #2". However, this does not fully capitalize on the known label words. To enhance this aspect, we propose the **Label-Aware Automatic Verbalizer for Multi-Label template ($LAAV_{ML}$)** to incorporate the label information into the training process, as illustrated in Figure 2.

First, for each label l_i, we compute the cosine similarity score, denoted as *cos*, between l_i and the remaining labels in $\mathcal{L}$. Then we construct $SL(l_i)$ which is the sequence of the other labels in $\mathcal{L}$ sorted by their cosine similarity scores with respect to l_i in ascending order. Based on Figure 2 (a), the sorted label of "joy" will be "fear", "optimism", and "love" in ascending order, for example.

Second, for each label l_i of a training input text x, we apply the following $LAAV_{ML}$ template to generate an actual training sample. Unlike the LAAV template in Equation 1, the $LAAV_{ML}$ template also uses other labels beyond l_i.

$$T_{l_i}(x) = [x]\ It\ was\ [SL(l_i|x)]\ [l_i]\ and\ \texttt{[MASK]} \tag{8}$$

where $SL(l_i|x)$ is the sequence $SL(l_i)$ that contains only the labels x has. Referring to Figure 2 (b), given the text input "Make sure it makes you #happy," which can be categorized into joy, optimism, and love, the text input is augmented into three additional samples. For the first sample, given l_i to be "joy", the $LAAV_{ML}$ template will be:

$$T_{joy}(x) = [x]\ It\ was\ [optimism\ love]\ [joy]\ and\ \texttt{[MASK]}$$

By substituting the input text x, the augmented text becomes "Make sure it makes you #happy. It was optimism love joy and [MASK]."

Finally, during the training phase presented in Figure 2 (c), the PLM is fine-tuned using two batches of training samples.

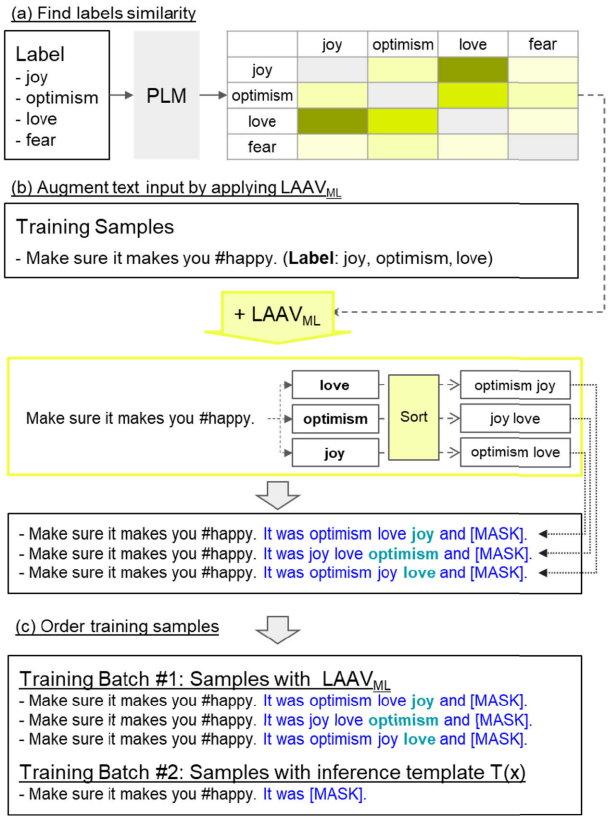

FIGURE 2. $LAAV_{ML}$: **a training sample generation process for PLAML.**

The first batch contains those generated by the $LAAV_{ML}$ template while the second batch contains those generated by the standard prompt-based learning template $T(x)$, which will also be used during inference to prevent the out-of-distribution problem. The specified training order (batch #1 followed by batch #2) is necessary to help the PLM properly predict a new text input. During the implementation, the order of training samples within each batch can be shuffled, but the shuffling does not extend across the two batches.

E. FINE-TUNING THE PLM

We fine-tune the PLM using the two training batches from the previous section. To do so, we obtain the predicted probability of each label using Equations 6 and 7 and apply the binary cross-entropy loss, which is more suitable for MLTC than the categorical-entropy loss [13]. Specifically, the loss associated to the training example $x_j \in \mathcal{D}_{train}$ is

$$L(x_j) = -\frac{1}{L}\sum_{i=1}^{L}[y_{ji}\log(p_{ji}) + (1-y_{ji})\log(1-p_{ji})] \quad (9)$$

Then the overall loss is the average of the loss from all examples in $\mathcal{D}_{train}$.

$$Loss = \frac{1}{n}\sum_{j=1}^{n} L(x_j) \quad (10)$$

Algorithm 1 Threshold Mechanism

Input: a label l_i, a validation dataset $\mathcal{D}_{val}$, and a fine-tuned PLM model M.
Output: the optimal threshold t_i for the label l_i

$T_{list} \leftarrow [\,]$ // An empty list to store the best thresholds
$best\_F1 \leftarrow 0$
for $t \in \{0.01, 0.02, \ldots 1.00\}$ **do**
  $F1 \leftarrow \text{COMPUTE\_F1}(\mathcal{D}_{val}, M, t)$
  if $F1 > best\_F1$ **then**
    $best\_F1 \leftarrow F1$
    $T_{list} \leftarrow [t]$
  else if $F1 = best\_F1$ **then**
    T_{list}.append(t)
  end if
end for
$t_i \leftarrow \text{AVERAGE}(T_{list})$
return t_i

F. DYNAMIC THRESHOLD MECHANISM

During validation and testing, we apply the template $T(x)$ to a given input text x_j and use the fine-tuned PLM to compute the probability of each label as in Equations 6 and 7. Generally, the predicted labels $\hat{y}_j$ for an input x_j are determined based on a threshold, t. In particular, $\hat{y}_{ji} = 1$ if $p_{ji} > t$; otherwise, 0. Many studies utilized a fixed thresholding approach, typically set at $t = 0.5$ [18], [19], for predicting each label. Meanwhile, Wei et al. [1] explored the variability in threshold values and concluded that the optimal fixed threshold may differ depending on each specific dataset. Despite this, the author continued to employ a fixed thresholding approach across all labels when evaluating the predicted probabilities for a given text input.

Outlined in Algorithm 1, we propose a dynamic threshold mechanism, in which distinct thresholds are computed individually for each label using the validation dataset $\mathcal{D}_{val}$. Note that, in our settings, the validation dataset serves exclusively for the thresholding mechanism, as we train all models for a fixed number of epochs. For each label in the classification task, we use a sequential search of threshold values, ranging from 0.01 to 1.00, and check the resulting F1 scores on $\mathcal{D}_{val}$. Ideally, the threshold value that yields the best F1 score should be employed. However, in few-shot settings, where the number of validation samples can be as limited as two samples per label, multiple threshold values may lead to identical F1 scores. To address this issue, we propose using the best thresholds averaging strategy. Specifically, the optimal threshold of the label is the average value of the thresholds that yield the highest F1 score of that label. Finally, we obtain a class-specific threshold t_i for each label l_i, and the predicted labels are defined using the class-specific thresholds as follows.

$$\hat{y}_{ji} = \begin{cases} 1, & \text{if } p_{ji} > t_i \\ 0, & \text{otherwise} \end{cases} \quad (11)$$

G. INFERENCE

Figure 1 (b) illustrates the inference process when applying our proposed framework. To elaborate, when we receive a new text input, denoted as x (e.g., "I personally prefer being happy."), we apply the template $T(x)$, which transforms the input into "I personally prefer being happy. It was `[MASK]`." This input is then tokenized before being fed into the fine-tuned PLM as described in Section III-E. The PLM outputs probability scores for all tokens, as defined in Equation 2. Next, we pass these probability scores to a verbalizer, which we obtain from Equation 6, and employ the dynamic threshold mechanism outlined in Section III-F to determine the labels for the given input.

IV. EXPERIMENTAL SETTINGS

A. DATASETS AND PRE-TRAINED MODELS

Table 1 shows an overview of the templates, labels, and other essential details for the three multi-label text classification datasets from the publicly available three languages we experimented with in this study. These datasets include SemEval-2018 Task 1: Affect in Tweets (English) [28], focused on emotion classification; Prachathai67k (Thai) [29], designed for news classification; and Filipino Dengue (Tagalog) [30], created for tweet classification. The pre-trained LMs used in this paper are the base versions of RoBERTa [11], Tagalog RoBERTa [31], and Wangchan-BERTa [32] for English, Tagalog, and Thai, respectively.

B. IMPLEMENTATION DETAILS

In the few-shot settings, we randomly selected 2, 4, 8, or 16 samples per label for both the training and validation splits. For constructing our training and validation datasets, we employed the minimum-including approach proposed by Hou et al. [33]. This method ensures that each label in the original dataset has a minimum of z samples in both the training and validation datasets, with z representing the number of shots specified for the particular setting. As we lack a sizable development dataset for hyperparameter optimization, we rely on insights from previous research to guide us in choosing the most suitable hyperparameters. All text inputs were limited to a maximum of 500 characters.

During the training process, we used Adam optimizer [34] with a learning rate of 1e-5 to optimize the loss function. We trained each model with 50 epochs and repeated the training process three times using different seeds to ensure robustness. To prevent overfitting, in addition to the Adam optimizer used for preventing fitting noise in the training data, we ensured that all three LMs used in our experiments incorporated dropout layers during each training iteration. Furthermore, our proposed dynamic threshold mechanism utilized validation datasets to provide an unbiased estimate of the thresholds based on unseen data after the training process.

We implemented our models using the PyTorch [35], Hugging Face transformers [36], and OpenPrompt [37] libraries. The training process was conducted on a Tesla P100 PCIe with 16 GB of memory.

TABLE 1. Details of the datasets along with their templates and labels.

SemEval-2018 Task 1: Affect in Tweets (English)	Number of Label	11
	Label Name	['anger', 'anticipation', 'disgust', 'fear', 'joy', 'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust']
	Test Samples Size	Total 7869: [1101, 425, 1099, 485, 1442, 516, 1143, 375, 960, 170, 153]]
	Inference Template: T(x)	[x] + ' It is about [MASK].'
	Manual Verb Adjustment	'pessimism' => 'pessim'
Prachathai 67k (Thai)	Number of Label	12
	Label Nam	['การเมือง' (politics), 'สิทธิมนุษยชน' (human rights), 'คุณภาพชีวิต' (quality of life), 'ต่างประเทศ' (foreign), 'สังคม' (social), 'สิ่งแวดล้อม' (environment), 'เศรษฐกิจ' (economy), 'วัฒนธรรม' (culture), 'แรงงาน' (labor), 'ความมั่นคง' (security), 'ไอซีที' (ICT), 'การศึกษา' (education)]
	Test Samples Size	Total 11027: [3842, 1151, 1127, 834, 789, 772, 519, 398, 350, 358, 292, 255]
	Inference Template: T(x)	[x] + 'เป็นข่าว [MASK]' English Translated: [x] + 'The news is [MASK].'
	Manual Verb Adjustment	'สิ่งแวดล้อม' (environment) => 'แวดล้อม' (surrounding) 'คุณภาพชีวิต' (quality of life) => 'ชีวิต' (life)
Filipino Dengue (Tagalog)	Number of Label	5
	Label Name	['absent', 'dengue', 'health', 'mosquito', 'sick']
	Test Samples Size	480 [76, 1, 252, 23, 128]
	Inference Template: T(x)	[x] + 'Ito ay tungkol sa [MASK].' English Translated: [x] + 'This is about [MASK].'

C. BASELINES

This subsection provides details of the baseline models used for comparative analysis. The **non-prompt-based learning baselines** include

One-vs-Rest: One-vs-Rest strategy utilizes embeddings from a PLM and feeds them into separate binary classifiers dedicated to each label, using a sigmoid activation function for predictions.

Full Fine-tune: Traditional fine-tuning method involves inputting the final `[CLS]` embedding into a classification layer with a sigmoid activation function to make predictions.

SpanEmo: A span-prediction task method proposed by Alhuzali and Ananiadou [18] which transforms an input text by adding label names in front of the text input and introduces a label-correlation-aware loss. We experimented with it using its official source code from GitHub.

Meanwhile, for **prompted-based learning baselines**, we included PTMLTC, NPPrompt, AMuLaP, and LAAV. We adapted the last three verbalizers, originally designed for multi-class classification, to suit MLTC. This adaptation involved modifying the initial loss function, making it

comparable to our proposed method. These approaches include

PTMLTC: A manual verbalizer proposed by Wei et al. [1] which uses label names as its verbalizer. We implemented it using the code provided by the OpenPrompt library [37]. Please note that the manual verb adjustment columns in Table 1 indicate that three label names from two datasets cannot be represented by a single token, therefore we instead selected their root words.

DiscreteV$_{\text{AMuLaP}}$: A modified version of AMuLaP [4] which constructs its discrete verbalizer using only the sample inputs and permits the verbalizer to select the same tokens across labels. We modified the verbalizer on top of the OpenPrompt library [37].

DiscreteV$_{\text{NPPrompt}}$: NPPrompt proposed by [5], which is a discrete verbalizer that uses embeddings from the PLM to select the closest words to be representative tokens. We implemented it using its official source code from GitHub.

DiscreteV$_{\text{LAAV}}$: A modified version of LAAV [8] which constructs its discrete verbalizer using sample inputs along with its label name and permits the verbalizer to select the same tokens across labels.

V. RESULTS

Within this section, our experiments are divided into four main parts. Section V-A involves the comparison of our proposed method with baseline models. This is followed by Sections (V-B, V-C, V-D), which include ablation studies to assess each of the three proposed techniques of PLAML individually. Finally, in Section V-E, we address potential challenges that may affect the wider use of our proposed techniques.

A. COMPARISON TO THE BASELINES

Table 2 presents a comprehensive comparison between the prompt-based learning approach and the traditional approach across various experimental settings. Notably, the prompt-based learning methods (PTMLTC, DiscreteV$_{\text{NPPrompt}}$, and DiscreteV$_{\text{LAAV}}$) consistently demonstrate superior performance when compared to the traditional approach (One-vs-Rest, Full Fine-tune, and SpanEmo) across all settings, with the performance gap widening as the number of training samples decreases. When examining the different baseline methods, it is evident that PTMLTC, known as a manual verbalizer, performs the best. The outcome proves that applying discrete verbalizers used in multi-class classification cannot be transferred to MLTC without appropriate modifications.

When comparing our proposed method PLAML with the baseline methods detailed section in Section IV-C, we found that our proposed approach consistently outperforms other baseline models across varying sample settings, including 4, 8, and 16 samples per label, across three datasets from three different languages. On average, our model enhances Macro-F1 scores by 1.68% absolute when compared to the best baseline, PTMLTC, across three datasets. The most significant improvement is observed in the Prachathai67k dataset, where our method boosts Macro-F1 scores by 2.39%. Additionally, PLAML improves the Macro-F1 scores on average compared to its original DiscreteV$_{\text{LAAV}}$ method by 4.26%.

In the context of the SemEval-2018 dataset, our result reveals that $LAAV_{ML}$ and TS-ILF have a comparable impact, with the removal of either of them yielding nearly identical outcomes. However, their efficacy diminishes notably when applying a fixed threshold of 0.5. In the Prachathai67k dataset, PLAML plays a more critical role in model accuracy, especially in larger datasets. The dynamic threshold mechanism only becomes important when dealing with smaller datasets, and TS-ILF has the least impact among the three

TABLE 2. Macro-F1 results along with their standard deviation in the parentheses comparing the proposed method, PLAML, and the baselines tested on three datasets. The best results are marked in bold. Indentation means that the experimental configuration of PLAML does not include the technique. Please note that the setting with the removal of the dynamic threshold means it instead uses the fixed threshold at 0.5.

Sample Size	2	4	8	16
SemEval-2018 Task 1: Affect in Tweets (English)				
One-vs-Rest	37.37 (2.94)	40.33 (2.25)	45.62 (0.53)	47.03 (1.65)
Full Fine-tune	35.31 (0.41)	40.05 (2.85)	44.58 (1.27)	48.79 (0.67)
SpanEmo	12.94 (8.44)	14.47 (6.01)	20.34 (1.05)	30.19 (7.75)
PTMLTC	46.33 (0.67)	48.31 (1.47)	51.36 (1.29)	52.65 (0.42)
DiscreteV$_{\text{AMuLaP}}$	34.92 (2.01)	39.92 (1.38)	46.42 (0.53)	49.13 (0.56)
DiscreteV$_{\text{NPPrompt}}$	45.68 (2.20)	49.04 (1.11)	51.13 (0.20)	52.60 (0.32)
DiscreteV$_{\text{LAAV}}$	45.20 (1.29)	47.77 (0.48)	50.57 (1.04)	52.29 (0.37)
PLAML	**47.73 (0.18)**	**49.16 (0.16)**	**52.51 (0.32)**	**54.06 (0.72)**
- $LAAV_{ML}$	46.46 (1.75)	48.33 (1.26)	50.19 (1.90)	52.73 (0.87)
- TS-ILF	45.87 (0.64)	48.43 (2.08)	50.58 (0.07)	51.95 (0.45)
- Dynamic Threshold	40.11 (2.18)	45.55 (1.41)	50.13 (1.79)	51.69 (0.73)
Prachathai67k (Thai)				
One-vs-Rest	31.32 (3.05)	33.01 (9.97)	37.43 (13.56)	47.88 (0.94)
Full Fine-tune	29.35 (0.76)	35.01 (0.07)	40.77 (1.97)	47.23 (0.86)
SpanEmo	24.17 (2.22)	31.98 (1.73)	39.81 (1.55)	46.66 (0.98)
PTMLTC	34.91 (2.61)	39.22 (1.66)	43.37 (2.53)	46.80 (2.17)
DiscreteV$_{\text{AMuLaP}}$	20.64 (0.42)	22.26 (0.87)	28.70 (0.87)	40.12 (0.60)
DiscreteV$_{\text{NPPrompt}}$	36.35 (3.97)	38.96 (1.83)	42.12 (3.33)	47.82 (0.30)
DiscreteV$_{\text{LAAV}}$	32.32 (1.02)	37.96 (2.02)	41.53 (0.89)	47.42 (1.04)
PLAML	36.82 (4.22)	**42.14 (2.09)**	**45.87 (1.35)**	**49.02 (0.67)**
- $LAAV_{ML}$	33.24 (3.63)	37.84 (0.83)	44.33 (1.89)	47.81 (1.91)
- TS-ILF	**38.13 (1.75)**	41.56 (2.12)	43.85 (1.50)	47.47 (0.91)
- Dynamic Threshold	30.23 (7.19)	39.82 (1.70)	43.98 (1.29)	48.27 (1.32)
Filipino Dengue (Tagalog)				
One-vs-Rest	35.25 (3.09)	44.74 (2.03)	49.03 (1.36)	58.85 (1.16)
Full Fine-tune	32.82 (4.72)	40.33 (1.22)	48.08 (1.63)	58.19 (1.55)
SpanEmo	25.37 (1.49)	36.15 (4.51)	46.07 (4.05)	53.38 (3.36)
PTMLTC	**51.79 (2.56)**	55.60 (2.48)	57.26 (2.15)	60.77 (1.22)
DiscreteV$_{\text{AMuLaP}}$	38.54 (6.52)	44.52 (5.35)	55.03 (5.40)	61.51 (1.23)
DiscreteV$_{\text{NPPrompt}}$	44.88 (3.48)	50.07 (1.19)	53.26 (2.49)	58.62 (2.60)
DiscreteV$_{\text{LAAV}}$	43.62 (6.76)	45.14 (9.01)	53.90 (2.15)	59.76 (1.15)
PLAML	51.61 (5.57)	**57.88 (2.87)**	**59.69 (2.43)**	**62.07 (0.89)**
- $LAAV_{ML}$	48.05 (5.57)	52.21 (5.31)	57.36 (2.74)	62.08 (0.87)
- TS-ILF	45.02 (2.29)	48.82 (1.33)	54.15 (3.03)	57.97 (1.44)
- Dynamic Threshold	49.78 (2.53)	55.08 (2.60)	58.66 (1.47)	61.14 (1.10)

components. Lastly, in the Filipino Dengue dataset, TS-ILF has the most substantial influence on model accuracy, closely followed by $LAAV_{ML}$.

Figure 3 provides a comprehensive comparison of Macro-Average ROC AUC scores, evaluating the model's ability to discriminate between positive and negative samples for each label. Among the baseline models (all bars except the yellow ones), DiscreteV$_{\text{NPPrompt}}$ excels in the SemEval-2018 (Figure 3 (a)) and the Filipino Dengue (Figure 3 (c)) datasets, while no clear winner emerges in the Prachathai67k dataset (Figure 3 (b)). This suggests that the verbalizers in different baselines exhibit varying performance on the Prachathai67k dataset.

As indicated in the yellow bar, our proposed model, PLAML, outperforms other baselines in both the SemEval-2018 and Prachathai67k datasets. However, in the Filipino Dengue dataset, PLAML performs comparably to DiscreteV$_{\text{NPPrompt}}$ and PTMLTC. A plausible explanation is the lower number of labels in the Filipino Dengue dataset. Consequently, the model does not have enough samples obtained from our sample generation process, $LAAV_{ML}$, to train effectively. Additionally, DiscreteV$_{\text{NPPrompt}}$ suggests all similar verbalizers to the label names used in PTMLTC, enabling faster learning under the same training epoch. Thus, it contributes to higher discriminative scores between labels, resulting in better Macro-Average ROC AUC scores. Overall, the results indicate that employing PLAML is the best option when considering measurement through Macro-Average ROC AUC score.

Additionally, Table 3 presents the top 8 (out of 32) representative tokens for the SemEval-2018 dataset as selected and ranked by different verbalizers. DiscreteV$_{\text{AMuLaP}}$ repeatedly selects the same words across different labels, including labels like "ME," which should not be associated with any of the labels. DiscreteV$_{\text{AMuLaP}}$ consistently chooses identical words across various labels, including labels like 'ME,' which should not be affiliated with any of the given labels. On the other hand, DiscreteV$_{\text{NPPrompt}}$, which utilizes embeddings from the PLM to select the closest words, tends to choose words that are similar to the label names. However, some selected words can be somewhat repetitive. For instance, the top 8 tokens of the label "love" are essentially variations of the word "love," such as "LOVE," "loved," and "loving." In contrast, both DiscreteV$_{\text{LAAV}}$ and PLAML tend to choose words that are closely related to the label names and carry significant meaning. For example, in DiscreteV$_{\text{LAAV}}$, the top tokens associated with the label "love" are "music" and "sacrifice," while in PLAML, they correspond to "laughter" and "forgiveness." Furthermore, these two verbalizers exhibit a more diverse set of word selections when compared to DiscreteV$_{\text{NPPrompt}}$ and DiscreteV$_{\text{AMuLaP}}$. Notably, PLAML assigns more refined weights to each token in comparison to DiscreteV$_{\text{LAAV}}$. For example, in DiscreteV$_{\text{LAAV}}$, the labels "joy" and "love" consider the word "peace" as one of their top words, while in PLAML, "peace" is assigned a low weight since it appears frequently across various labels, resulting in more refined word choices.

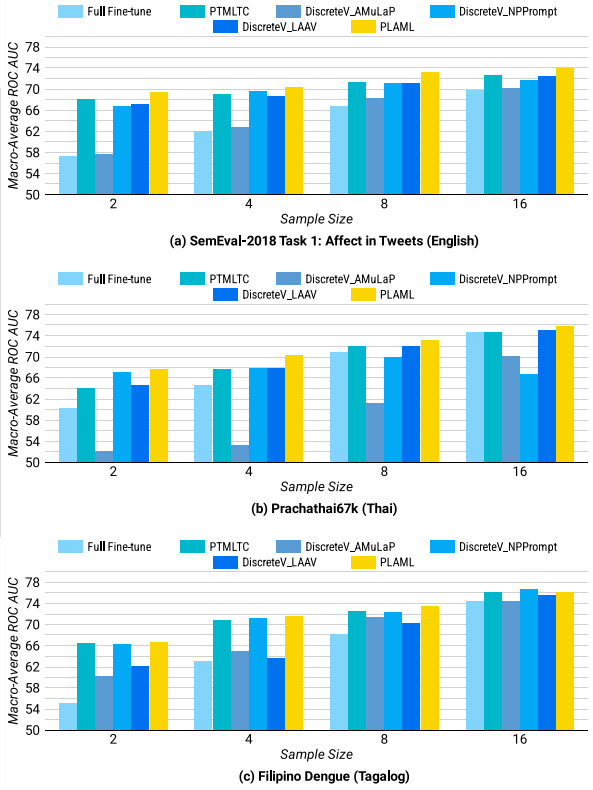

FIGURE 3. Macro-Average ROC AUC results comparing the proposed method, PLAML, and the baselines tested on three datasets.

B. EFFECTS OF TS-ILF

In this section, our objective is to examine the impact of different weight approaches in comparison to our proposed TS-ILF including "Uniform Weight," which assigns uniform scores to all labels, and "Token Probability," which utilizes probability scores from the PLM.

According to Table 4, our weight-based approach, detailed in III-F demonstrates superior performance with at least 1.50% increases in Marco-F1 in most settings compared to the two alternative methods This superiority is particularly noticeable across all datasets, with the most improvement observed in the Filipino Dengue dataset.

In Figure 4, we experimented with varying a different number of tokens to represent each label, denoted as k. This is because the different values of k affect the token representation for verbalizer and the ILF component within the TS-ILF. The result indicates that a higher number of tokens used to represent each label is correlated with an increase in Macro-F1, often reaching its peak value at 32, as indicated by the yellow bar. As a practical recommendation, when working with a new dataset, we advise testing a range of k values, as different k values can lead to variations in accuracy.

TABLE 3. Comparison of the verbalizer based on the top 8 tokens having the highest probability scores out of the 32 tokens in the 4-shot setting using the SemEval-2018 dataset.

Label Name	DiscreteV$_{AMuLaP}$	DiscreteV$_{NPPrompt}$	DiscreteV$_{LAAV}$	Proposed Method: PLAML
anger	'over', 'time', 'due', 'enough', 'to', 'her', 'here', 'perfect'	'anger', 'angry', 'Anger', 'fury', 'frustration', 'indignation', 'rage', 'angered'	'violence', 'hurt', 'division', 'conflict', 'aggression', 'rebellion', 'defiance', 'vengeance'	'grief', 'revenge', 'resentment', 'pain', 'anger', 'despair', 'passion', 'fear'
anticipation	'luck', 'change', 'speed', 'freedom', 'style', 'winning', 'nothing', 'fun'	'anticipation', 'anticipating', 'expectation', 'excitement', 'apprehension', 'anticipate', 'anticipated', 'preparation'	'time', 'risk', 'action', 'planning', 'expectations', 'waiting', 'desire', 'timing'	'waiting', 'preparation', 'expectation', 'anticipation', 'growth', 'possibility', 'uncertainty', 'patience'
disgust	'there', 'due', 'enough', 'to', 'her', 'here', 'karma', 'perfect'	'disgust', 'disgusted', 'dismay', 'disdain', 'indignation', 'displeasure', 'frustration', 'outrage'	'outrage', 'contempt', 'embarrassment', 'fury', 'remorse', 'humiliation', 'disdain', 'disgust'	'contempt', 'humiliation', 'disbelief', 'resentment', 'hate', 'disappointment', 'pain', 'despair'
fear	'times', 'her', 'history', 'control', 'heart', 'perspective', 'survival', 'nothing'	'fear', 'fears', 'Fear', 'feared', 'fearful', 'afraid', 'fearing', 'scared',	'terror', 'survival', 'vulnerability', 'intimidation', 'prejudice', 'doubt', 'worry', 'pride'	'hate', 'uncertainty', 'rage', 'pain', 'anger', 'intimidation', 'worry', 'pride'
joy	'heart', 'style', 'winning', 'perspective', 'survival', 'fun', 'attitude', 'him'	'joy', 'delight', 'joyful', 'pleasure', 'Joy', 'happiness', 'enjoyment', 'excitement'	'health', 'light', 'strength', 'discovery', 'healing', 'harmony', 'peace', 'purpose'	'honesty', 'family', 'fun', 'happiness', 'trust', 'passion', 'discovery', 'healing'
love	'home', 'power', 'redemption', 'everyone', 'history', 'heart', 'perfect', 'winning'	'love', 'LOVE', 'Love', 'loves', 'loved', 'loving', 'adore', 'hate'	'music', 'sacrifice', 'kindness', 'redemption', 'community', 'power', 'loss', 'peace'	'laughter', 'forgiveness', 'faith', 'passion', 'loss', 'loyalty', 'respect', 'understanding'
optimism	'freedom', 'style', 'ME', 'winning', 'perspective', 'survival', 'fun', 'attitude'	'optimism', 'optimistic', 'enthusiasm', 'skepticism', 'hopeful', 'cynicism', 'uncertainty', 'upbeat'	'energy', 'promise', 'dreams', 'resilience', 'change', 'opportunity', 'purpose', 'inspiration'	'opportunity', 'optimism', 'faith', 'happiness', 'change', 'patience', 'gratitude', 'commitment'
pessimism	'simple', 'priorities', 'honesty', 'guts', 'change', 'communication', 'control', 'ME'	'pessim', 'pessimistic', 'essim', 'optim', 'nihil', 'skept', 'Optim', 'optimism'	'stress', 'weakness', 'isolation', 'negativity', 'gloom', 'cynicism', 'delusion', 'doubt'	'gloom', 'delusion', 'regret', 'anger', 'confusion', 'despair', 'hope', 'greed'
sadness	'speed', 'control', 'karma', 'ME', 'nothing', 'attitude', 'him', 'choice'	'sadness', 'sorrow', 'sad', 'melancholy', 'saddened', 'loneliness', 'disappointment', 'bitterness'	'suffering', 'tears', 'reflection', 'guilt', 'sorrow', 'loneliness', 'longing', 'change'	'loneliness', 'anxiety', 'disappointment', 'acceptance', 'desperation', 'joy', 'hope', 'suffering'
surprise	'people', 'there', 'times', 'everyone', 'to', 'here', 'history', 'perfect'	'surprise', 'surprises', 'surprised', 'Surprise', 'surprising', 'shock', 'unexpected', 'shocking'	'challenge', 'learning', 'surprise', 'danger', 'wonder', 'error', 'magic', 'mystery'	'challenge', 'learning', 'surprise', 'error', 'intrigue', 'inspiration', 'horror', 'anticipation'
trust	'relationships', 'consistency', 'teamwork', 'communication', 'freedom', 'karma', 'ME', 'winning'	'trust', 'trusts', 'Trust', 'trusting', 'trust', 'trusted', 'distrust', 'mistrust'	'support', 'relationship', 'responsibility', 'connection', 'sharing', 'communication', 'integrity', 'transparency'	'sharing', 'community', 'confidence', 'belief', 'friendship', 'patience', 'compassion', 'courage'

C. EFFECTS OF LAAV$_{ML}$

The *LAAV*$_{ML}$ uses a template that comprises label arrangements placed atop a label-aware template, as in Equation 1. To construct the most effective *LAAV*$_{ML}$ template, we conducted two experiments. The first experiment involved exploring the order of labels and their influence on the Macro-F1 score, presented in Table 5. To elaborate on the "Single Label" approach, we first considered the example from Figure 2: "Make sure it makes you #happy." In this approach, this single example will be augmented into three separate samples which include only "love", "optimism", and "joy" in the first, second, and third samples, respectively. For "Random Ordered", we simply randomized the order of labels without considering their similarity. The results indicate that our "Sorted" approach, which takes label similarity into account, yields slightly better results compared to other methods. One potential explanation is that the sorted order of labels encapsulates the relationship among the labels and could help the model better learn the classification task.

TABLE 4. Comparing the effects of different weighting strategies tested on three datasets, the Macro-F1 results and their standard deviation in the parentheses are presented. The best results are marked in bold.

Sample Size	2	4	8	16
SemEval-2018 Task 1: Affect in Tweets (English)				
Uniform Weight	45.87 (0.64)	48.43 (2.08)	50.58 (0.07)	51.95 (0.45)
Token Probability	44.38 (0.73)	48.57 (1.58)	49.34 (1.97)	51.79 (0.57)
PLAML: TS-ILF	**47.73 (0.18)**	**49.16 (0.16)**	**52.51 (0.32)**	**54.06 (0.72)**
Prachathai67k (Thai)				
Uniform Weight	**38.13 (1.75)**	41.56 (2.12)	43.85 (1.50)	47.47 (0.91)
Token Probability	37.45 (3.58)	40.63 (1.30)	43.92 (2.00)	46.92 (0.33)
PLAML: TS-ILF	36.82 (4.22)	**42.14 (2.09)**	**45.87 (1.35)**	**49.02 (0.67)**
Filipino Dengue (Tagalog)				
Uniform Weight	45.02 (2.29)	48.82 (1.33)	54.15 (3.03)	57.97 (1.44)
Token Probability	47.85 (1.38)	49.95 (2.94)	53.01 (3.22)	58.95 (2.09)
PLAML: TS-ILF	**51.61 (5.57)**	**57.88 (2.87)**	**59.69 (2.43)**	**62.07 (0.89)**

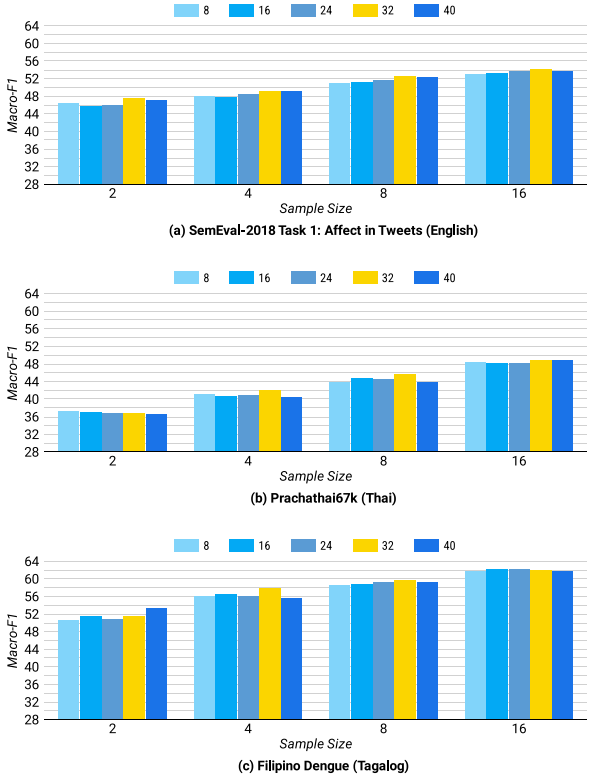

FIGURE 4. Macro-F1 results when using PLAML with a different number of tokens to represent each label varying from 8, 16, 24, 32, and 40.

In Table 6, our second experiment aims to explore other separators for the given sorted labels. Our choices of separators include "and", "or", " " (single space), and "," (comma). The result shows that a single space (employed in PLAML) is the most effective choice achieving the best results in nine settings and the second best in the remaining three settings.

D. THRESHOLD MECHANISM

Our proposed dynamic threshold mechanism returns a different threshold for each label using the validating samples as described in Section III-F. To evaluate our proposed threshold, we conducted another experiment by comparing it with fixed thresholding approach ranging from 0.1 to 0.5. Based on Figure 5, the dynamic thresholding approach consistently surpasses the fixed thresholding approach in most scenarios, as indicated by the yellow bar. In contrast, the performance of the fixed thresholding approach exhibits variability when applied to different settings and datasets. In contrast, our approach adapts its threshold based on the validating dataset, making it more robust to varying settings. Notably, setting a fixed threshold at 0.2 yields the best results for the SemEval-2018 dataset (Figure 5 (a)). The Prachathai67k dataset (Figure 5 (b)) performs the best with a fixed threshold of 0.3 while applying a fixed threshold of 0.5 appears to be most effective on the Filipino Dengue dataset (Figure 5 (c)).

TABLE 5. Comparing the effect of label ordering approaches used to construct $LAAV_{ML}$ tested on three datasets, the Macro-F1 results, along with their standard deviation in the parentheses, are presented. The best results are marked in bold.

Sample Size	2	4	8	16
SemEval-2018 Task 1: Affect in Tweets (English)				
Single Label	45.18 (2.66)	47.00 (1.65)	48.84 (1.26)	50.58 (0.66)
Random Ordered	45.81 (1.39)	48.90 (1.39)	51.38 (0.53)	53.06 (0.60)
PLAML: Sorted	**47.73 (0.18)**	**49.16 (0.16)**	**52.51 (0.32)**	**54.06 (0.72)**
Prachathai67k (Thai)				
Single Label	36.19 (3.61)	40.22 (1.87)	45.78 (0.94)	48.81 (0.46)
Random Ordered	36.66 (4.51)	40.70 (2.20)	45.01 (0.98)	48.48 (0.84)
PLAML: Sorted	**36.82 (4.22)**	**42.14 (2.09)**	**45.87 (1.35)**	**49.02 (0.67)**
Filipino Dengue (Tagalog)				
Single Label	49.74 (6.24)	56.00 (2.80)	58.05 (1.88)	**62.19 (0.63)**
Random Ordered	50.94 (5.88)	56.38 (2.77)	59.62 (1.83)	61.68 (0.79)
PLAML: Sorted	**51.61 (5.57)**	**57.88 (2.87)**	**59.69 (2.43)**	62.07 (0.89)

TABLE 6. Comparing the effect of different separators used to construct $LAAV_{ML}$ tested on three datasets, the Macro-F1 results, along with their standard deviation in the parentheses, are presented. The best results are marked in bold.

Sample Size	2	4	8	16
SemEval-2018 Task 1: Affect in Tweets (English)				
And	46.69 (1.59)	49.12 (0.63)	50.95 (0.40)	53.15 (0.33)
Comma	46.03 (1.25)	47.64 (2.10)	51.86 (0.83)	52.74 (0.73)
Or	46.33 (1.40)	48.28 (1.32)	51.58 (0.88)	52.93 (0.50)
PLAML: Space	**47.73 (0.18)**	**49.16 (0.16)**	**52.51 (0.32)**	**54.06 (0.72)**
Prachathai67k (Thai)				
And	36.43 (3.68)	40.15 (2.28)	44.17 (0.49)	43.44 (4.96)
Comma	**37.22 (3.26)**	40.75 (2.62)	43.10 (3.05)	48.00 (0.57)
Or	35.31 (5.33)	40.30 (3.08)	43.87 (0.51)	48.00 (0.48)
PLAML: Space	36.82 (4.22)	**42.14 (2.09)**	**45.87 (1.35)**	**49.02 (0.67)**
Filipino Dengue (Tagalog)				
And	50.41 (5.57)	56.68 (2.87)	58.49 (2.43)	60.87 (0.89)
Comma	50.79 (5.92)	56.71 (3.14)	**59.85 (1.47)**	61.03 (0.80)
Or	51.52 (6.19)	56.54 (2.57)	58.94 (2.64)	**62.52 (0.74)**
PLAML: Space	**51.61 (5.57)**	**57.88 (2.87)**	59.69 (2.43)	62.07 (0.89)

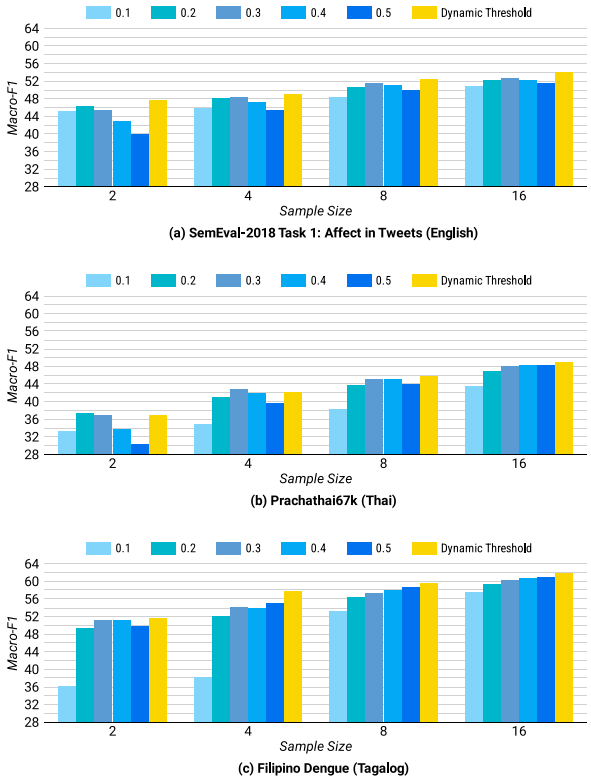

FIGURE 5. Macro-F1 results when using PLAML with different threshold approaches tested on three datasets.

TABLE 7. F1 results, along with their corresponding thresholds in parentheses, are presented based on the 4-shot setting using the SemEval-2018 dataset. This comparison evaluates different choices of thresholding approaches for each label, with the best results highlighted in bold.

Label Name	Fix Threshold (t:0.5)	Dynamic Threshold
anger	67.58	**68.18 (t:0.6714)**
anticipation	24.50	**31.20 (t:0.2900)**
disgust	68.02	**70.21 (t:0.4058)**
fear	44.91	**46.92 (t:0.5001)**
joy	76.92	**79.16 (t:0.4452)**
love	**52.87**	47.77 (t:0.6911)
optimism	65.83	**69.53 (t:0.3658)**
pessimism	29.42	**30.10 (t:0.0400)**
sadness	59.78	**63.20 (t:0.1120)**
surprise	13.73	**15.74 (t:0.0933)**
trust	15.34	**16.74 (t:0.6555)**
Macro-F1	47.12	**48.98**

Furthermore, Table 7 shows F1 scores of the SemEval-2018 dataset under a 4-shot setting, consistent with the configuration used in Table 3. The dynamic thresholding approach selects optimal thresholds ranging from 0.04 to 0.6 and achieves better F1 scores across all labels compared to the fixed threshold at 0.5. This highlights the importance of adopting the dynamic thresholding approach to few-shot MLTC because it helps the LM to be more robust to variations in label characteristics and label distribution across different settings.

E. LIMITATIONS

Our primary focus was on enhancing the word selection process for each label within a predefined prompt template. The application of a tunable continuous template or multiple discrete templates, not explored in this work, may have the potential to reduce input ambiguity and improve prompt-based learning results. Additionally, due to resource constraints, we conducted experiments using only the base version of the PLMs. Experimentation with larger PLMs can be investigated in the future.

VI. CONCLUSION

In this paper, we proposed PLAML, a framework designed to improve the overall efficiency of the prompt-based learning approach for MLTC in few-shot settings. The framework includes *(i)* TS-ILF that weights each token within a verbalizer based on its probability and its corresponding frequency across label set, *(ii)* $LAAV_{ML}$ that uses the label name to augment more training samples, and *(iii)* a dynamic threshold mechanism that assigns each label with a different threshold. The experiments demonstrate the effectiveness of PLAML in few-shot settings across three datasets, including three distinct languages that vary from low-resource to high-resource. In the future, we plan to explore the application of PLAML in more complex natural language processing tasks and apply the method to multilingual LMs.

REFERENCES

[1] L. Wei, Y. Li, Y. Zhu, B. Li, and L. Zhang, "Prompt tuning for multi-label text classification: How to link exercises to knowledge concepts?" *Appl. Sci.*, vol. 12, no. 20, p. 10363, Oct. 2022.

[2] Y. Wang, Y. Wang, Z. Peng, F. Zhang, L. Zhou, and F. Yang, "Medical text classification based on the discriminative pre-training model and prompt-tuning," *Digit. HEALTH*, vol. 9, Jan. 2023, Art. no. 20552076231193213.

[3] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Comput. Surv.*, vol. 55, no. 9, pp. 1–35, Jan. 2023, doi: 10.1145/3560815.

[4] H. Wang, C. Xu, and J. McAuley, "Automatic multi-label prompting: Simple and interpretable few-shot classification," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2022, pp. 5483–5492.

[5] X. Zhao, S. Ouyang, Z. Yu, M. Wu, and L. Li, "Pre-trained language models can be fully zero-shot learners," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, 2023, pp. 15590–15606.

[6] T. Schick, H. Schmid, and H. Schütze, "Automatically identifying words that can serve as labels for few-shot text classification," in *Proc. 28th Int. Conf. Comput. Linguistics*, 2020, pp. 5569–5578.

[7] T. Gao, A. Fisch, and D. Chen, "Making pre-trained language models better few-shot learners," in *Proc. 59th Annu. Meeting Assoc. Comput. Linguistics 11th Int. Joint Conf. Natural Lang. Process.*, 2021, pp. 3816–3830.

[8] T. Thaminkaew, P. Lertvittayakumjorn, and P. Vateekul, "Label-aware automatic verbalizer for few-shot text classification," 2023, *arXiv:2310.12778*.

[9] K. W. Church, "Word2Vec," *Nat. Lang. Eng.*, vol. 23, no. 1, pp. 155–162, 2017.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[11] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[12] H. Tsai, J. Riesa, M. Johnson, N. Arivazhagan, X. Li, and A. Archer, "Small and practical BERT models for sequence labeling," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 3632–3636.

[13] J. Liu, W.-C. Chang, Y. Wu, and Y. Yang, "Deep learning for extreme multi-label text classification," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2017, pp. 115–124.
[14] T. Schick and H. Schütze, "Exploiting cloze-questions for few-shot text classification and natural language inference," in *Proc. 16th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2021, pp. 255–269.
[15] P. Yang, X. Sun, W. Li, S. Ma, W. Wu, and H. Wang, "SGM: Sequence generation model for multi-label classification," in *Proc. 27th Int. Conf. Comput. Linguistics*, 2018, pp. 3915–3926.
[16] A. Pal, M. Selvakumar, and M. Sankarasubbu, "Multi-label text classification using attention-based graph neural network," 2020, *arXiv:2003.11644*.
[17] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, Jan. 2020.
[18] H. Alhuzali and S. Ananiadou, "SpanEmo: Casting multi-label emotion classification as span-prediction," in *Proc. 16th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2021, pp. 1573–1584.
[19] R. Song, Z. Liu, X. Chen, H. An, Z. Zhang, X. Wang, and H. Xu, "Label prompt for multi-label text classification," *Appl. Intell.*, vol. 53, no. 8, pp. 8761–8775, Apr. 2023.
[20] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: Pre-training text encoders as discriminators rather than generators," 2020, *arXiv:2003.10555*.
[21] Q. Wang, S. Dai, B. Xu, Y. Lyu, Y. Zhu, H. Wu, and H. Wang, "Building Chinese biomedical language models via multi-level text discrimination," 2021, *arXiv:2110.07244*.
[22] A. Holtzman, P. West, V. Shwartz, Y. Choi, and L. Zettlemoyer, "Surface form competition: Why the highest probability answer isn't always right," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2021, pp. 7038–7051.
[23] K. Hambardzumyan, H. Khachatrian, and J. May, "WARP: Word-level adversarial ReProgramming," in *Proc. 59th Annu. Meeting Assoc. Comput. Linguistics 11th Int. Joint Conf. Natural Lang. Process.*, 2021, pp. 4921–4933.
[24] K. Ji, Y. Lian, J. Gao, and B. Wang, "Hierarchical verbalizer for few-shot hierarchical text classification," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, 2023, pp. 2918–2933.
[25] G. Cui, S. Hu, N. Ding, L. Huang, and Z. Liu, "Prototypical verbalizer for prompt-based few-shot tuning," in *Proc. 60th Annu. Meeting Assoc. Comput. Linguistics*, 2022, pp. 7014–7024.
[26] S. Hu, N. Ding, H. Wang, Z. Liu, J. Wang, J. Li, W. Wu, and M. Sun, "Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification," in *Proc. 60th Annu. Meeting Assoc. Comput. Linguistics*, 2022, pp. 2225–2240.
[27] J. Ramos, "Using TF-IDF to determine word relevance in document queries," in *Proc. 1st Instructional Conf. Mach. Learn.*, 2003, vol. 242, no. 1, pp. 29–48.
[28] S. Mohammad, F. Bravo-Marquez, M. Salameh, and S. Kiritchenko, "SemEval-2018 task 1: Affect in tweets," in *Proc. 12th Int. Workshop Semantic Eval.*, 2018, pp. 1–17.
[29] (2019). *Prachathai67k*. [Online]. Available: https://github.com/PyThaiNLP/prachathai-67k
[30] J. Christian Blaise Cruz and C. Cheng, "Establishing baselines for text classification in low-resource languages," 2020, *arXiv:2005.02068*.
[31] J. Christian Blaise Cruz and C. Cheng, "Improving large-scale language models and resources for Filipino," 2021, *arXiv:2111.06053*.
[32] L. Lowphansirikul, C. Polpanumas, N. Jantrakulchai, and S. Nutanong, "WangchanBERTa: Pretraining transformer-based Thai language models," 2021, *arXiv:2101.09635*.
[33] Y. Hou, Y. Lai, Y. Wu, W. Che, and T. Liu, "Few-shot learning for multi-label intent detection," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 14, pp. 13036–13044.
[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[35] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 8024–8035.
[36] T. Wolf et al., "HuggingFace's transformers: State-of-the-art natural language processing," 2019, *arXiv:1910.03771*.
[37] N. Ding, S. Hu, W. Zhao, Y. Chen, Z. Liu, H. Zheng, and M. Sun, "OpenPrompt: An open-source framework for prompt-learning," in *Proc. 60th Annu. Meeting Assoc. Comput. Linguistics, Syst. Demonstration*, 2022, pp. 105–113.

THANAKORN THAMINKAEW received the first B.Sc. degree in statistics from the College of Science, and the second B.Sc. degree in economics from the College of the Liberal Arts, The Pennsylvania State University, State College, PA, USA, in 2018. He is currently pursuing the M.Sc. degree in computer science with the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Thailand. His research interests include machine learning, deep learning, and natural language processing.

PIYAWAT LERTVITTAYAKUMJORN received the Ph.D. degree from the Department of Computing, Imperial College London, U.K., in 2022. Currently, he is a full-time Research Scientist with Google, USA. His research interests include natural language processing, explainable AI, and human–AI collaboration.

PEERAPON VATEEKUL (Member, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Miami (UM), Coral Gables, FL, USA, in 2012. Currently, he is an Associate Professor with the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Thailand. His research interests include machine learning, data mining, deep learning, text mining, big data analytics, natural language processing, and applied deep learning techniques in various domains, such as healthcare, geoinformatics, hydrometeorology, and energy trading.