Cascading Models of CNN and GRU with Autoencoder Loss for Precipitation Forecast in Thailand

Fuenglada Manokij[1], Kanoksri Sarinnapakorn[2], and Peerapon Vateekul[3]

ABSTRACT: It is a crucial task to accurately forecast precipitation, especially in Thailand, since the forecasts are used in flood prevention and agricultural planning. However, the performance of prior attempts has been limited due to three challenging issues. First, the rain pattern in each region can be different. It is hard to get an accurate rainfall forecast with a single model for the whole country. Second, there is an imbalance issue with data, where most rainfall is zero because Thailand has a short rainy season. Third, the predicted rainfall is underestimated since moderate and heavy rainfall cases rarely occur. This paper proposes a deep learning model that cascades the CNN and GRU models to forecast rainfall in Thailand and creates a model for each region based on local rain behaviors. CNN is specialized for classifying rain and non-rain events. In the CNN stage, the imbalanced issue is alleviated by applying "focal loss". GRU is responsible for forecasting rainfall. Its predicted range is lifted using "autoencoder loss". The experiment was conducted on hourly rainfall data between 2012 and 2018 obtained from Hydro-Informatics Institute, Thailand. The results show that our model outperforms SARIMA and other ensemble models in terms of RMSE for most regions in Thailand.

Keywords: Precipitation Prediction, Deep Learning, CNN, GRU, Autoencoder

DOI: 10.37936/ecti-cit.2021153.240957

Article history: received May 31, 2020; revised August 3, 2020; accepted December 8, 2020; available online November 12, 2021

1. INTRODUCTION

Accurate precipitation prediction is one of the especially important tasks in Thailand since it is applied in various activities, such as flood prevention and agricultural planning. To capture the rainfall amount, there are many rain-gauge stations installed across the whole country. Apart from the rainfall amount, these stations collect other information, including, temperature, pressure, humidity, and cumulative hourly rainfall.

In the meteorology field, there are many studies of rainfall prediction. One approach is based on a physics model, e.g. WRF and WRFROMS [1]. However, this approach has a high computational cost, and the configuration requires manual adjustment by experts. Another approach is based on machine learning and statistical techniques, e.g., ARIMA [2,3], machine learning models [4,5,6,7,8], and deep learning models [9,10,11,12,13]. Among these techniques, deep learning has shown the most promising results.

However, the previous work does not show sufficient prediction accuracy due to three main challenges. First, the rainfall amount varies in each region. It is not easy to predict rainfall accurately using a single universal model for the whole country. Second, the proportions of rain and non-rain events are

[1,3]The authors are with Chulalongkorn University Big Data Analytics and IoT Center (CUBIC), Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand., E-mail: 6071024421@student.chula.ac.th and peerapon.v@chula.ac.th

[2]The author is with Hydro-Informatics Institute (Public Organization) Ministry of Higher Education, Science, Research and Innovation, Bangkok, Thailand., E-mail: kanoksri@hii.or.th

[3]Corresponding author: peerapon.v@chula.ac.th

highly imbalanced because of the short rainy season each year. This results in most data points being zero (non-rain). For example, the proportion of non-rain periods in our dataset is 95.39% while the proportion of rain periods is 4.61% in the northern region of Thailand. Third, most models cannot forecast moderate and heavy rain events since both rain events rarely occur with only an 0.006% occurrence rate on average in all regions of Thailand. Heavy rainfall can be considered as an extreme event or anomaly, which cannot be captured by any traditional machine learning techniques.

The autoencoder can be applied to detect extreme events by learning and generating a regular raining pattern [17,18,19]. If the prediction result is different from the output of the autoencoder (i.e., normal pattern), this is considered the sign of an extreme event.

In this paper, we present a novel deep learning framework to forecast rainfall in Thailand. The model is specifically designed to address all of the above issues. First, models are generated separately for each region, so that they can capture local rainfall behavior. Second, the imbalanced data issue (rain and non-rain events), where most data values are zero, can be alleviated by using a cascading model consisting of (i) Convolutional Neural Network (CNN) [20] and (ii) Gated Recurrent Unit (GRU) [21]. CNN will classify rain and non-rain events, and GRU will predict rainfall amounts for the raining periods classified by CNN. The focal loss [22] is then employed to handle a class imbalance issue. Lastly, as extreme events are usually underestimated due to the scarcity of moderate and heavy rainfall events (anomaly events), the concept of anomaly detection is applied through an introduction of a new loss called "autoencoder loss". To further improve performance, exogenous factors such as temperature, pressure, and humidity are included in our model. The experiment was conducted on hourly rainfall data (2012 -2018) recorded at rain-gauge stations by the Hydro-Informatics Institute (HII).

This paper is organized as follows. Section 2 presents the literature review. Section 3 introduces the proposed model. Section 4 describes the experimental setup. Section 5 presents the experimental results. Finally, the last section is the conclusion.

2. LITERATURE REVIEW

This section provides summary information about related works that can be divided into two groups. The first group are models to forecast rainfall amount, and the second group are models to detect heavy rain events (anomalies).

2.1 Rainfall prediction model

From our literature survey, there are two main approaches to a rainfall forecasting model based on rain-gauge information: (*i*) a statistical approach and (*ii*) a machine learning approach.

The first approach uses a statistical model as a baseline, e.g., ARIMA and SARIMA. In 2013, Wang et al. [3] presented a Seasonal Autoregressive Integrated Moving Average (SARIMA) model to deal with precipitation data in Shouguang city, China. SARIMA is an enhanced ARIMA model that uses seasonal factors. Their experiment showed that SARIMA could predict rainfall accurately. Therefore, SARIMA was chosen as a baseline in our paper.

The second approach is based on machine learning techniques including deep learning networks. In 2009, Hung et al. [7] used an ANN model with various features, such as wind, temperature, rainfall in the previous steps, and rainfall from neighboring rain-gauge stations, to predict rainfall in the Bangkok area. In 2014, Liu et al. [8] presented a deep neural network model along with external factors (i.e., temperature, dew points, MSLP, and the wind speed) to predict weather change in the next 24 hours in Hong Kong. Both of these works indicated that a weather forecasting model can be improved by including external factors.

Many recent works on the second approach showed that the neural networks and deep learning models can provide promising results. In 2016, Hernández et al. [11] proposed an ensemble method using an autoencoder and multi-layer perceptron models called "AE-MLP" to forecast daily accumulated rainfall. For this model, an encoder aimed to extract nonlinear features, and a multi-layer perceptron aimed to forecast rainfall in the next time step. In 2017, Qiu et al. [12] presented a multi-task convolutional neural network to forecast short-term rainfall. Their extraction concept was the same as that used in [11], but they used a CNN model that considered feature correlation between weather features and neighboring sites' information. The model outperformed both statistical and conventional deep learning models. In 2019, Manokij et al. [13], our prior work proposed a cascading model between CNN and GRU to forecast rainfall in Thailand. The purpose of this model was to tackle the class imbalance of rain events found in Thailand's rainfall dataset. CNN was used to classify rain and non-rain events and GRU was used to forecast rainfall amounts based on rain events data only. This method could filter non-rain events that overwhelmed the rainfall dataset, making up 95.39% of the total data, and increased GRU performance to learn only rain patterns. Although the model could capture low rainfall events quite well, it still showed low accuracy for moderate and heavy rainfall events.

There were some prior works that tried to find a relationship between particulate matter pollution (PM 2.5 and 10) and weather conditions. In 2018, Kliengchuay et al. [14] showed that there was a relationship between PM factors and many weather fea-

tures, such as temperature, rainfall, pressure, carbon monoxide (CO), and ozone (O3). In 2019, Joseph [15] presented an IoT system based on the raspberry pi board to record all air factors. Then, [16] used collected information from [15] to predict PM 2.5. However, all of these works focused on weather monitoring (current situation) and did not propose an algorithm to forecast rainfall amounts. In our work, the main contribution focuses on proposing a modern algorithm to forecast rainfall. Data used in this research was collected from rain-gauge stations installed across all regions in Thailand by HII between 2012 and 2018. The data contains rainfall amounts along with three exogenous variables: temperature, humidity, and pressure.

2.2 Anomaly detection model

From previous studies in anomaly detection, a deep learning model was applied to detect anomaly values in many ways. An autoencoder model, which is an unsupervised model created by a neural network, is comprised of two models called "Encoder" and "Decoder" as shown in Fig. 1. The encoder model tries to learn and extract a pattern from the input, while the decoder model tries to generate the output from the information in the encoder model.

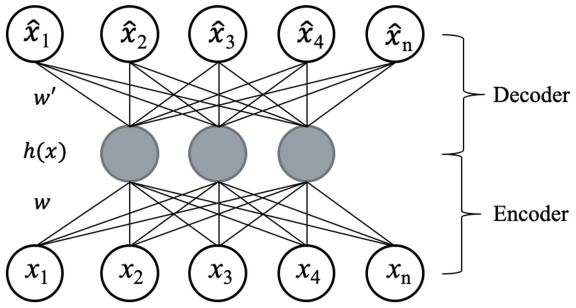

Fig. 1: *The basic structure of autoencoder model.*

In Fig. 1, the input x is fed into the hidden layer of $h(x)$ via an activation function represented as $a(x)$ to capture a high-level representation. To produce output, $h(x)$ is fed into the decoder layer to reconstruct the output $\hat{x}$ via the activation function of $a(x)$. The sigmoid function is typically used as the activation function to ensure the scope of the output values range from 0 to 1.

The Autoencoder processes can be expressed as Equations (1) and (2).

$$h(x) = f(a(x)) = Sigmoid(Wx + b) \tag{1}$$

$$\hat{x} = g(a(x)) = Sigmoid(W'x + b') \tag{2}$$

The purpose of an autoencoder model is to reconstruct input without noise. If we apply the model to predict an abnormal range of data, it will lead to a high reconstruction error.

In 2015, Xia et al. [18] introduced an autoencoder model for anomaly detection tasks. In 2016, Malhotra et al. [19] presented a long short-term memory (LSTM) autoencoder to detect anomalies in time-series data collected from sensors. Because a traditional autoencoder is used for non-sequential data, LSTM is employed to handle time-series data which is sequential data. The LSTM encoder will learn a fixed-length vector representation of input and the LSTM decoder will map the representation to reconstruct a sequential output using the current hidden state and the previous time step value. Reconstruction error is used to detect anomalies. If the error is higher than a threshold, the output value is flagged as an anomaly value.

In this paper, the model is improved by incorporating external factors and applying reconstruction loss from an autoencoder. SARIMA and AE-MLP are considered our baselines.

3. PROPOSED MODEL

In our previous work [13], we found imbalanced data issues in Thailand's rainfall dataset. If we use a single deep learning model for the whole dataset, the model will give bad performance in detecting rainfall amounts because the dataset is overwhelmed by non-rain periods. A cascading model that combined the classification and regression models was proposed to tackle this problem. However, our previous model still could not detect high rainfall amounts. To make model able to detect extreme rainfall, we propose a cascading model with autoencoder loss to boost the prediction performance.

3.1 Cascading model of CNN and GRU [13]

Our cascading model has two components: a classification model and a regression model. The architecture of the cascading model is shown in Fig. 2.

3.1.1 Classification model

The concept of classification is to filter rain periods by detecting rain or no rain in the next time step because the regression model should only be fed input with only rain periods. We create input as a matrix with dimension $F \times T$, where F is the number of features, and T is the number of previous time steps, and feed it into the two CNN layers. CNN will consider all features at each step before extracting non-linear features that are important to predict rain or no rain. Architecture and hyperparameters in CNN layers are set up similarly to those in two previous studies [12,23]. As the dataset has many non-rain periods, we apply focal loss [22] with a sigmoid function to address the class imbalance. Focal loss is one of the loss functions that helps a model

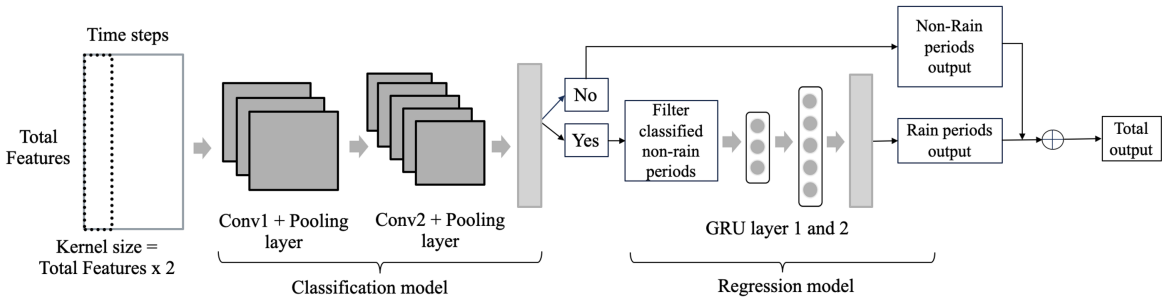

Fig.2: *Architecture of our cascading model.*

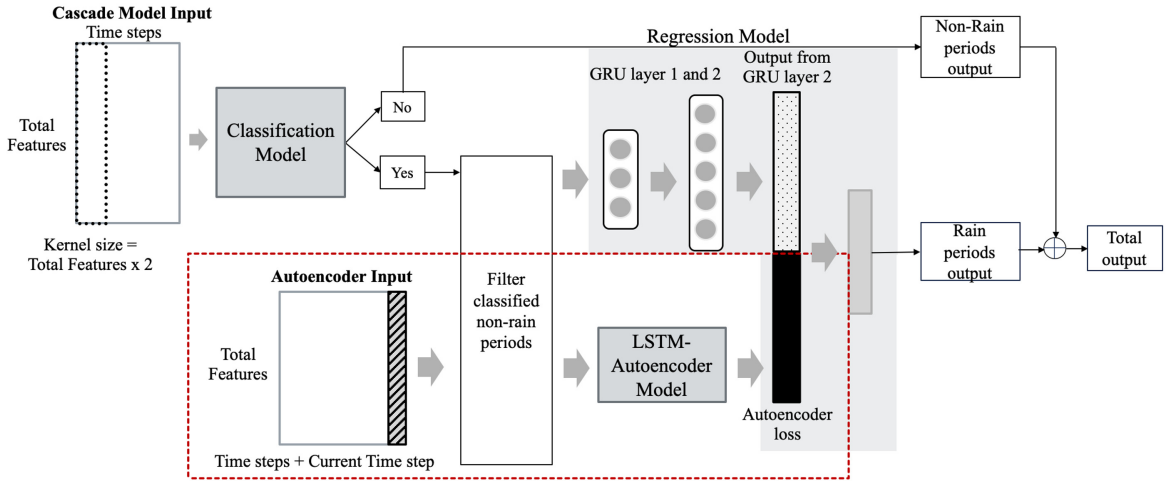

Fig.3: *Architecture of our cascading model with autoencoder loss (in the large dashed rectangle).*

detect a rare class. In this context, the rare class is the rain class. The important hyperparameters are the weighting factor α and the modulating factor γ. For non-rain class examples, we assume the rainfall amount in the next time steps to be 0 mm. The focal loss function is shown in (3),

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t) \tag{3}$$

t is the observed time step, and p_t is the probability value from loss computation at t. α_t is a weighting factor for addressing class imbalance with a range between 0 and 1, and γ is a modulating factor to adjust the weight for easy examples with a value equal or greater than 0.

3.1.2 Regression model

We use a dataset that has only rain events, which are results from the classification model, to allow the regression model to learn rain periods and forecast rainfall amounts. This approach captures future rainfall in each step more accurately than when feeding the whole set of data to the regression model at once. The dimension of the input vector is composed of total data in rain periods, the number of previous time steps (we use 12-time steps), and the number of total features. Similar to our previous work, this model has two Gated Recurrent Unit (GRU) layers that consider all features from the lookback time steps in a rain period's scope, and then predicts the rainfall amount in the next time step. We combine the rainfall amounts from the non-rain class and the rain class to compute total RMSE.

3.2 Cascading model with autoencoder loss

Our cascading model in section 3.1 is improved by adding an autoencoder model to generate a reconstruction error value and feed it as one of the inputs in the regression model, as shown in Fig. 3. This helps alleviate the underestimation issue of rainfall amounts seen in the previous model.

The autoencoder model is widely used for anomaly detection. The autoencoder aims to reconstruct an output from an input that has the same characteristics as the original input. In anomaly detection, the autoencoder model predicts data points that have an abnormal distance from others called "outliers". Then, the model will evaluate the error between the actual value and the predicted value. This is called "reconstruction error". For an outlier data point, the model should generate a high reconstruction error. By applying this anomaly detection method, the autoencoder model will create reconstruction errors

from rainfall data.

From the model structure shown in Fig. 3, CNN in our cascading model with autoencoder loss has the same structure as CNN in a normal cascading model, but the structure of GRU is improved. First of all, we construct a new input at the current time step because we want the model to learn anomaly values at observation hours. Only rain periods are extracted as autoencoder input by using the classification model to filter rain class positions. Since the autoencoder aims to reconstruct common rain events (light ones), we train the autoencoder with rainfall data smaller than 5 mm only. Most rainfall values are in the range of 0.2 – 5 mm. as shown in rainfall histogram in Fig. 4. The data handling process before feeding into the autoencoder model can be seen in Fig. 5.

The structure of the autoencoder model is shown in Fig. 6. There are four LSTM layers, including the encoder and decoder parts. A sequence of inputs is fed into the encoder part mapping data into an embedding vector. The decoder part converts the embedding vector back into a sequence of inputs again [24]. For the model input, there are 14 features with a window of 13 time steps (12 look-back data points and one current data point where the interval of each data point is one hour). Then, we feed the input into the encoder model, which has two LSTM layers. We use more hidden nodes in the first layer than in the second layer because we need a model to learn the regular pattern and compress data in the hidden nodes. The output from the encoder model is a high level representation of the original input. After that, we pass the output from the encoder model as input to the decoder model via the repeat vector layer, which creates an output shape which is the same as the input shape. The objective of the decoder model is to generate new data by learning from compressed data. Thus, the number of hidden nodes in the third layer is less than the number in the last layer to rectify and generate the last output. The autoencoder model in Fig. 6 (the large dashed box in Fig. 3) is trained by using the rainfall amounts in the normal range to learn the common rain pattern. We then use the Mean Squared Error (MSE) at each data point, which is a measure of the difference between the actual rainfall and the predicted rainfall, to create the reconstruction error. Fig. 7 shows an example of reconstruction errors in the southern region for both normal and abnormal ranges. The orange line is rainfall amount in millimeters (mm.), and the blue dashed line is reconstruction error in MSE. The reconstruction error tends to be high for any abnormal rainfall amount (> 5 mm.)

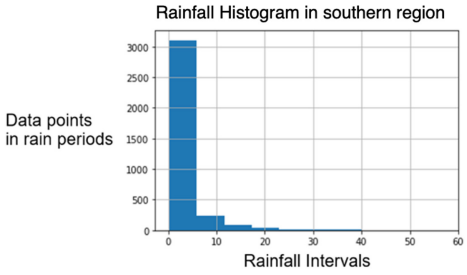

Fig.4: *The histogram of rainfall in rain periods from the southern region.*

For the last part in Fig. 3, we combine the forecasted rainfall amount (top part) and the reconstruction error or autoencoder loss (bottom part). The reconstruction error is flattened into a vector and then merged with an output from GRU layer 2. The merged vector is fed into a dense layer in the regression model. In the regression model, the reconstruction error represents event triggering, and it helps the model predict higher rainfall.

4. EXPERIMENTAL SETUP

4.1 Dataset

We obtained rainfall data from 469 rain-gauge stations located around Thailand that was collected by the Hydro-Informatics Institute and denoted it as the HII dataset. For the sake of experiment, one station from each region is selected which met conditions below:

- It contains data from years between 2012 and 2018.
- It must contain less than 30% missing values.
- It must not contain more than 30% of adjacent rainfall data points with values of 0.2 mm., which is the minimum value of rainfall, since that implies that the site has suffered data loss or has incomplete data.

The rain-gauge transactions are recorded every hour. There are 61,368 data points in total. The data was divided with years 2012 to 2016 as a training set, 2017 as a validation set, and 2018 as a testing set. There are four exogenous factors: temperature (Celsius), humidity (mbar), pressure (%), and rain amount (mm.).

4.2 Data preprocessing

This topic contains three parts: data interpolation, feature extension, and data normalization.

4.2.1 Data interpolation

Due to faulty network communication or system failures, it is not unusual to find missing values in data collected from rain-gauge stations. It is crucial to perform different interpolation techniques to handle missing values based on rainfall behavior in each region and season.

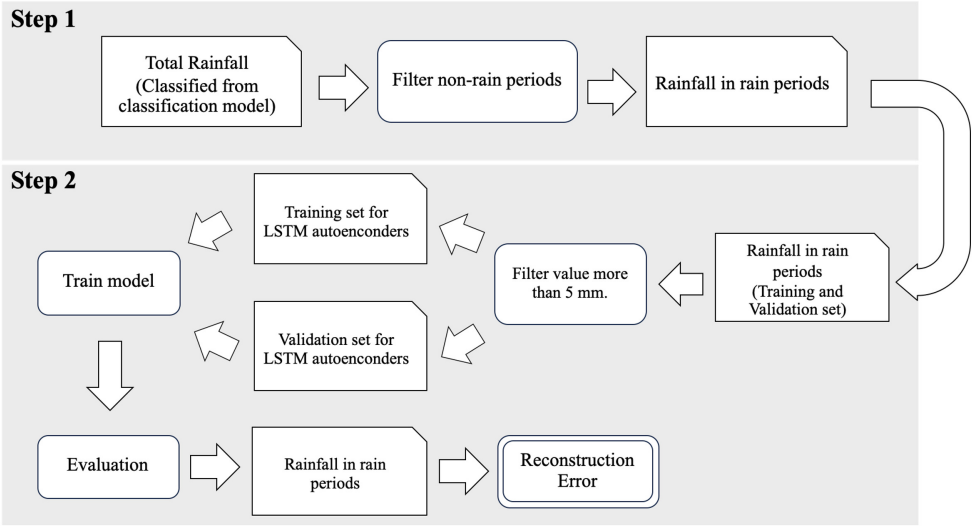

Fig.5: *The process of data handling before feeding it to the autoencoder model.*

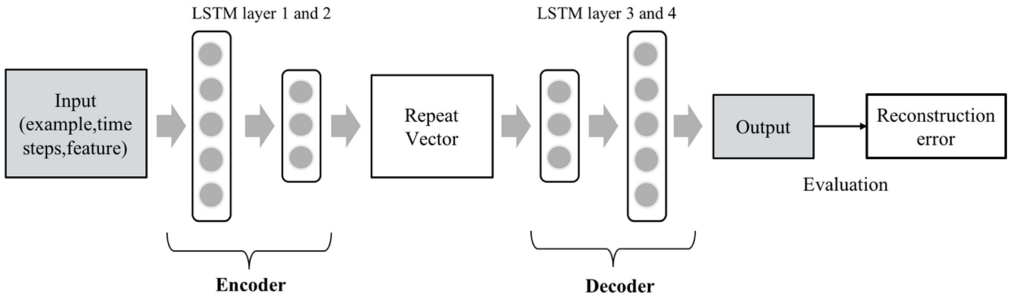

Fig.6: *Structure of LSTM autoencoder.*

Fig.7: *Graph of reconstruction errors from testing set in the southern region. The left y axis represents the actual rainfall (orange line), and the right y axis shows the reconstruction error (blue line).*

Southern region: Mostly the southern region of Thailand has rain throughout the year because it is near the sea and is influenced by monsoons. From previous statistics of rainfall amount per year, we found that the accumulated rainfall was the highest among all regions in Thailand. Therefore, data from any rain-gauge station in the southern region is interpolated by using an average of value in the previous time step before a missing value and value in the next time step after the missing value.

Other regions: Thailand is located 15 degrees above the Equator, which causes Thailand to have a hot and humid climate. Due to this climate, Thailand has a long period of warm weather and less rain, especially in Thailand's northern and northeastern regions. Accumulated rainfall in the rainy season is higher than in other seasons. Therefore, we performed the same interpolation technique as we did with the south of Thailand for the rainy season and filled in zero values for other seasons.

4.2.2 Feature extension

There are two kinds of factors added to our model. First, we create rain or non-rain classes based on rainfall amounts. If the rainfall amount is at least 0.2 mm., it belongs to the rain class, otherwise it belongs to the non-rain class. Second, statistical features [12] are added to boost prediction performance. These include the maximum, the sum, and the average of rainfall amounts within three-time intervals from the current time point (i.e., last 1, 2, and 3 hours).

14 features are added in our experiment: 4 weather features, 1 rain/non-rain flag, and 9 statistical features.

4.2.3 Data normalization

It is necessary to convert data from different units to be on the same scale before feeding them to deep learning networks. In this work, we use standardization to transform rainfall data. Standardization changes the data distribution to have a mean of 0 and a standard deviation of 1. The standardization formula is given in Equation (4).

$$Z_i = \frac{x_i - \mu}{\sigma} \tag{4}$$

z_i is a standardized value, i is the observed time step, x_i is the original value, μ is the mean of the training set, and σ is the standard deviation of the training set.

4.3 Performance evaluation

To evaluate our regression model's performance, we used Root Mean Square Error (RMSE) to measure the error between the actual rainfall amount and the predicted rainfall amount. We used RSME to evaluate 3 periods: overall periods, rain periods, and non-rain periods. The RMSE equation is shown in (5).

$$RMSE\ Error\ = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2} \tag{5}$$

i is the observed time step, y_i is the predicted value, x_i is the observed value, and n is the number of examples.

We use the F1 score and confusion matrix to evaluate classification performance. We use both metrics to tune the model to detect a rain class (true positive) and not to identify a non-rain class as a rain class (false positive). The formula for the F1 score is shown in (6), and the confusion matrix is shown in Table 1.

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6}$$

Precision is the fraction of relevant instances among the actual instances, and *Recall* is the fraction of total relevant results that are correctly classified.

Table 1: *Confusion matrix.*

		Actual Value	
		Positives	Negatives
Predicted Value	Positives	True Positive (TP)	False Positive (FP)
	Negatives	False Negative (FN)	True Negative (TN)

4.4 Hyperparameters and training

We used the same parameters in the classification and regression models from the traditional cascading model for our cascading model with autoencoder loss.

In the autoencoder model, we use four LSTM layers. The first layer and last layer of the model use more hidden nodes than the second and third layers. In the regression problem, we use two GRU layers and a dense layer with a dropout layer [25] that prevents an overfitting issue. The solver is Adaptive Moment Estimation (Adam) optimization [26] in both models. The optimal hyperparameters are summarized in Table 2.

Table 2: *Summary of hyperparameters.*

Model	Layer	Hyperparameters
Classification model	CNN Layer 1	8 filters, kernel size = Features x 2
	CNN Layer 2	16 filters, kernel size = 1 x 3
	Max Pooling layer	Pool size = 1 x 2
	Fully Connected	512
LSTM Autoencoder	LSTM first and last layer	64
	LSTM second and third layer	16
Regression model	GRU layer 1 and 2	32,128
	Dense layer	256
	Dropout	0.25

Our experimental model was built on one GPU card from NVIDIA, a Tesla K20Xm, a CPU with 16 cores, and 64 GB of RAM supported by HII. All experiments were implemented in Python3 and used the Keras API.

5. EXPERIMENTAL RESULTS

In this section, we show that our model improved both classification and regression performance. A good classification technique should correctly classify rain classes (rain and non-rain), and its performance

Table 3: *The proportions between non-rain & rain events and F1 scores for rain class from the classification models.*

Region	Support class			F1 score for rain class		% increase F1 score in rain class
	Non-rain	Rain	Rain class proportion (%)	Binary cross-entropy loss	Focal loss	
Central	8,321	421	4.82%	31.25%	**38.07%**	+21.82%
East	7,946	796	9.11%	31.56%	**49.75%**	+57.64%
North	7,689	1,055	12.07%	51.44%	**58.10%**	+12.95%
Northeast	8,339	403	4.61%	37.67%	**44.30%**	+17.60%
South	7,843	899	10.28%	28.72%	**45.67%**	+59.02%

is measured in terms of the F1 score. For the regression method, a good model should forecast rainfall amounts accurately (with small errors), and its performance is measured in terms of RMSE.

SARIMA is considered as a baseline method. The cascading model that combines CNN and GRU is denoted as CNN-GRU, and our proposed model is denoted as CNN-AE-GRU. Furthermore, we compare our proposed model with another ensemble deep learning method called autoencoder with multilayer-perceptron. This model was introduced in [11] and denoted as AE-MLP. The results will be discussed in three main parts: 1) classification result, 2) forecasted rainfall amount, and 3) forecasted rainfall amount during rain periods.

5.1 Classification result

Table 3 shows the classification performance of both cascading models based on the F1 score. Since they use the same classification module, the results are the same. We compare two different loss functions: binary cross- entropy (baseline) and focal loss (ours). The F1 score of the model with focal loss outperforms that with the binary cross-entropy loss in all regions. The southern region has the biggest improvement of 59.02% from 28.72% to 45.67%, while the northern region has the lowest improvement of 12.95% from 51.44% to 58.10%.

The result shows that the focal loss handles the imbalance class issue. It increases the weight of the rain class and reduces the weight (importance) of the non-rain class resulting in an improvement of the F1 score in the rain class.

5.2 Forecasted rainfall amount

To evaluate the regression performance, we use RMSE to compare our proposed model with the baseline models, which are SARIMA and AE-MLP. The overall results are shown in Table 4, and the details of rain and non-rain periods are reported in Tables 5 and 6.

In Table 4, both cascading models based on a deep learning approach show fewer errors than a statistical model, SARIMA, in most regions except the eastern region. Also, the model with our autoencoder loss (CNN-AE-GRU) is the winner, and it outperforms both the traditional cascading model (CNN-GRU) and the previous ensemble deep learning model (AE-MLP). When we compare the results between CNN-GRU and CNN-AE-GRU, the biggest improvement is shown in the eastern region, with a 1.50% improvement from 1.8328 to 1.8053 in terms of RMSE. The lowest improvement is shown in the northeastern region with only a 0.09% reduction from 0.9843 to 0.9834 in terms of RMSE.

Table 4: *The overall RMSE performance in mm. for each region. Boldface is the winner. The % diff is a comparison between CNN-AE-GRU and CNN-GRU.*

Region	**SARIMA**	**AE-MLP**	**CNN-GRU**	**CNN-AE-GRU**	**Diff. (%)**
Central	0.9267	0.9301	0.9173	**0.9141**	0.35%
East	1.8071	1.8227	1.8328	**1.8053**	1.50%
North	0.9100	0.9085	0.9068	**0.9027**	0.45%
Northeast	0.9915	0.9982	0.9843	**0.9834**	0.09%
South	1.5731	1.5621	1.5636	**1.5559**	0.49%

Table 5: *The RMSE of the rain periods for each region. Bold face is the winner. The % diff is a comparison between CNN-AE-GRU and CNN-GRU.*

Region	**SARIMA**	**AE-MLP**	**CNN-GRU**	**CNN-AE-GRU**	**Diff. (%)**
Central	4.1844	4.1863	4.1237	**4.1194**	0.10%
East	5.9080	5.9346	5.9846	**5.9060**	1.31%
North	2.5681	2.5472	2.5293	**2.5276**	0.07%
Northeast	4.5559	4.5227	4.5189	**4.4985**	0.45%

Table 6: *The RMSE of the non-rain periods for each region. Bold face is the winner. The % diff is a comparison between CNN-AE-GRU and CNN-GRU.*

Region	SARIMA	AE-MLP	CNN-GRU	CNN-AE-GRU	Diff. (%)
Central	**0.1345**	0.1488	0.1538	0.1386	9.88%
East	0.3101	0.3563	0.3283	**0.3024**	7.89%
North	**0.1851**	0.2196	0.2390	0.2232	6.61%
Northeast	0.1722	0.2367	**0.1699**	0.1893	-11.41%
South	0.3524	0.3365	0.3294	**0.3097**	5.98%

The results of rain and non-rain periods are shown in Tables 5 and 6. For the rain periods in Table 5, the autoencoder loss improved the performance in all regions. The most significant improvement is in the eastern region with 1.31% reduction from 5.9846 to 5.9060 in terms of RMSE. For the non-rain periods in Table 6, the autoencoder loss improved the prediction results in almost all regions, except for the northern region. The most significant improvement is in the central region with 9.88% from 0.1538 to 0.1386 in terms of RMSE. In the northeastern region, the autoencoder model's performance dropped since it tries to predict a higher rainfall amount, while this region barely has any moderate and heavy rain events.

5.3 Forecasted rainfall amount during rain periods

In this section, we also focus on rainfall levels in rain periods. The rain periods are divided into three levels: light, moderate, and heavy rain. The criterion for splitting rainfall is shown below:

- Light rain : rainfall between 0.2 and 10 mm.
- Moderate rain : rainfall between 10 and 35 mm.
- Heavy rain : rainfall greater than 35 mm.

Table 7 shows the forecasting performance in terms of RMSE for all three levels (light, moderate, and heavy). CNN-AE-GRU has better performance than CNN-GRU in the light rainfall in all regions (except the eastern area) and in the moderate-heavy rainfall in most regions. Since the autoencoder has learned the raining patterns in normal circumstances, the difference between the predictions of the autoencoder model and the actual values is called "autoencoder loss" and shows signs of extreme raining events. This can guide a regression model to increase the predicted rainfall amount resulting in better performance in moderate and heavy rain events.

It is interesting to note that for the light rain events, the autoencoder loss can improve the RMSE in almost all regions. The central region has the biggest RMSE reduction of 2.84% from 2.0634 to 2.0047. For the moderate rain events, it shows improvements in the eastern, northeastern, and southern regions. The heavy rain cases result is quite similar to the moderate cases, except in the northeastern region. Since the autoencoder model can capture the signs of high rainfall amounts on moderate and heavy rainfall events by detecting these from the reconstruction error, its results can help the regression model increase the predicted rainfall amounts for these events. The most successful case is in the southern region where the autoencoder loss improves the performance in all levels of rainfall since there is more and heavier rain in this region. The last column in Table 7 shows that the autoencoder loss increases the forecasting value as it is higher in all regions.

Fig. 8 - 15 show the comparison graphs between the actual rainfall amounts and the forecasted rainfall amounts from the SARIMA, AE-MLP, CNN-GRU, and CNN-AE-GRU models. We choose the results from observation sites in the northeastern region, which has very scarce rainfall and where most values are zero, and the southern region that has rain throughout the year. The graphs show that the SARIMA model in Fig. 8 (the northeastern region) and Fig. 12 (the southern region) can detect higher rainfall compared to the others. However, AE-MLP in Fig. 9 and CNN-GRU in Fig. 10 (the northeastern region) with AE-MLP in Fig. 13 and CNN-GRU in Fig. 14 (the southern region) can capture rainfall amounts more accurately than SARIMA for the light rain events. The prediction graphs from AE-MLP are similar to the graphs from CNN-GRU. However, the graphs from both models (Fig. 8 and Fig. 11) show that our normal cascading model can forecast rainfall amounts only for the light rain events. It barely forecasts any rainfall amount in the abnormal range (> 5 mm.). Subsequentially adding autoencoder loss, the graphs from CNN-AE-GRU in Fig. 11 (the northeastern region) and Fig. 15 (the southern region) show that most of the forecasted rainfall amounts are higher than other deep learning models predicted and most values are greater than 5 mm.

Based on the RMSE, CNN-AE-GRU can reduce error values from CNN-GRU in light and moderate rain periods, but it still cannot detect rainfall in heavy rain periods because of the small size of the data points.

6. CONCLUSION

This paper presents a deep learning framework to forecast rainfall in Thailand. We propose two cascading models to handle different issues. First, CNN is used to classify rain and non-rain cases, while addressing an excessive amount of non-rain cases using "focal loss". This loss function can help our model detect rain cases by reducing the weight of the non-rain class and increasing important weight of the rain class. Second, GRU is used for a regression of rainfall and is tailored to improve moderate and heavy rain-

Table 7: *The RMSE performance from rain periods for each of the rainfall levels, and maximum predicted rainfall for each region. Bold face is the winning method.*

Region	Model	Rain periods	Light rain periods	Moderate rain periods	Heavy rain periods	Maximum Predicted rainfall
Central	CNN-GRU	4.1237	2.0634	**16.8093**	38.4000	5.6868
	CNN-AE-GRU	**4.1194**	**2.0047**	16.8441	38.4000	**6.2195**
	Difference (%)	+0.10%	+2.84%	-0.21%	0%	+9.37%
East	CNN-GRU	5.9846	**2.5674**	17.0068	45.9130	13.3213
	CNN-AE-GRU	**5.9060**	2.6801	**16.9915**	**41.5423**	**20.1177**
	Difference (%)	+1.31%	-4.39%	+0.09%	+9.52%	+51.02%
North	CNN-GRU	2.5293	1.4368	**13.1917**	**40.6212**	7.7188
	CNN-AE-GRU	**2.5276**	**1.4140**	13.2681	40.9122	**12.1607**
	Difference (%)	+0.07%	+1.59%	-0.58%	-0.72%	+57.55%
Northeast	CNN-GRU	4.5189	2.2448	16.1864	**39.7282**	5.7750
	CNN-AE-GRU	**4.4985**	**2.2299**	**16.0280**	40.2697	**7.8639**
	Difference (%)	+0.45%	+0.66%	+0.98%	-1.36%	+36.17%
South	CNN-GRU	4.7778	2.2503	16.3808	39.4288	9.9966
	CNN-AE-GRU	**4.7560**	**2.2429**	**16.3389**	**39.3347**	**13.6371**
	Difference (%)	+0.46%	+0.39%	+0.26%	+0.24%	+36.42%

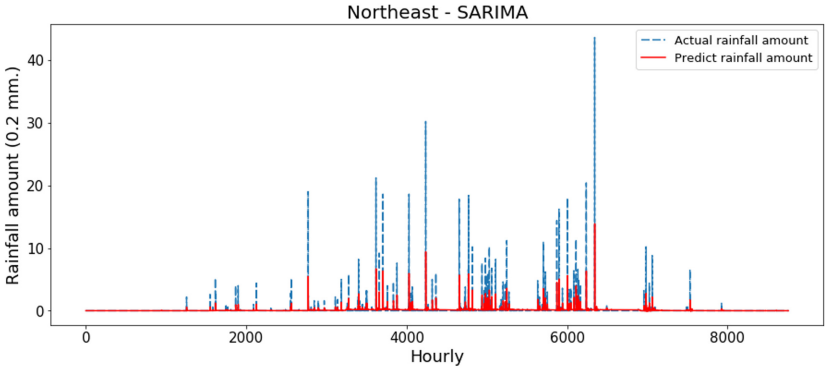

Fig.8: *Predictive rainfall using SARIMA for the northeastern region.*

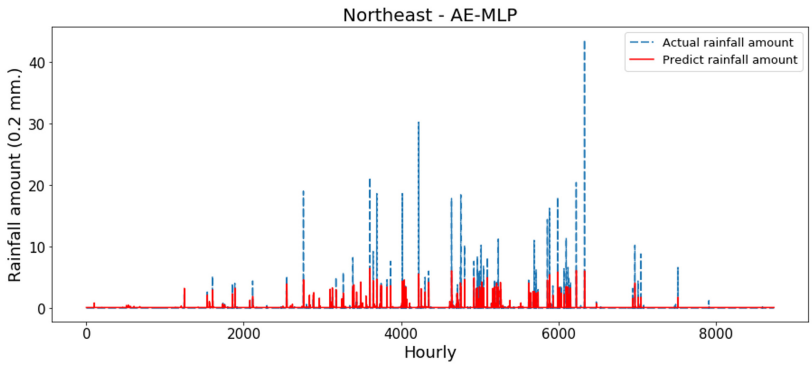

Fig.9: *Predictive rainfall using AE-MLP for the northeastern region.*

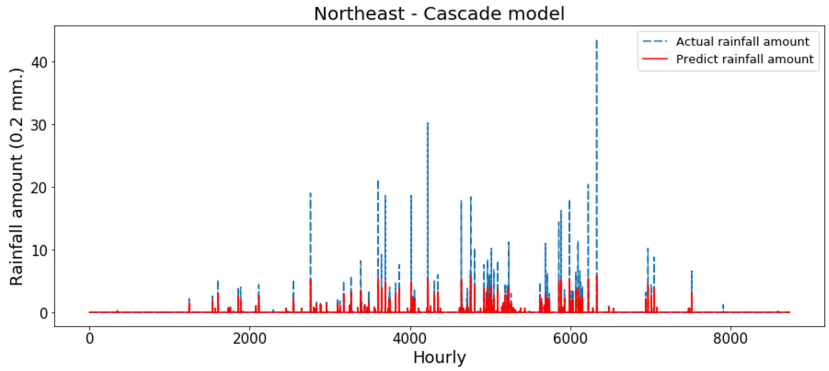

Fig.10: *Predictive rainfall using CNN-GRU for the northeastern region.*

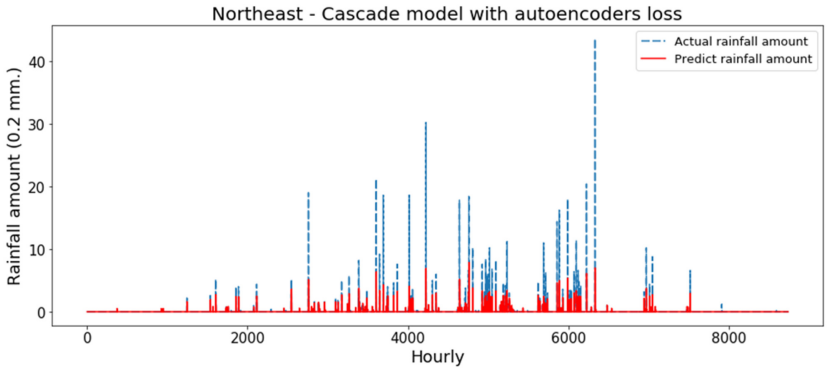

Fig.11: *Predictive rainfall using CNN-AE-GRU for the northeastern region.*

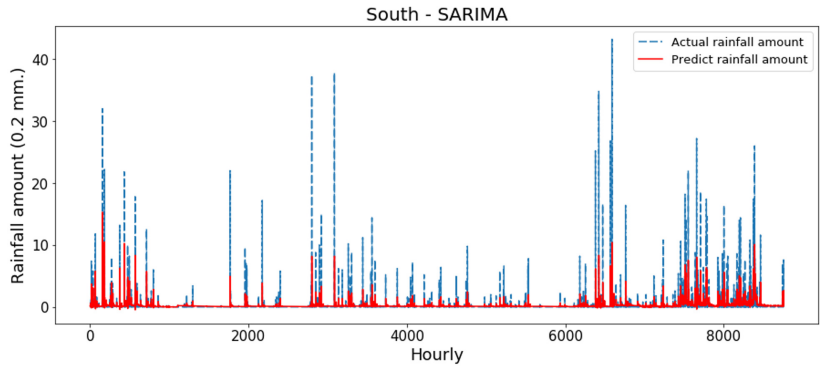

Fig.12: *Predictive rainfall using SARIMA for the southern region.*

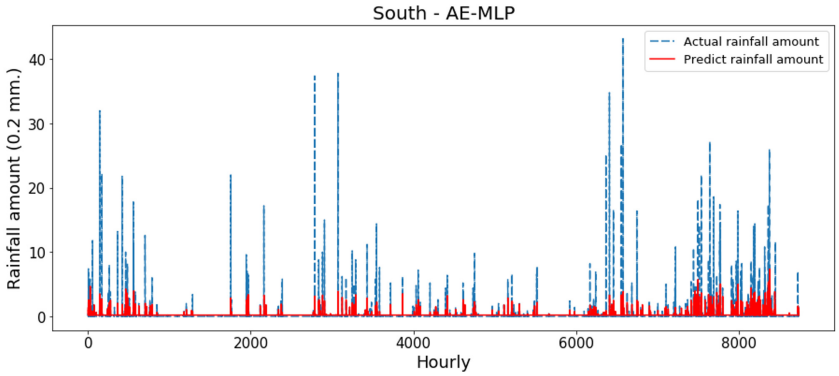

Fig.13: *Predictive rainfall using AE-MLP for the southern region.*

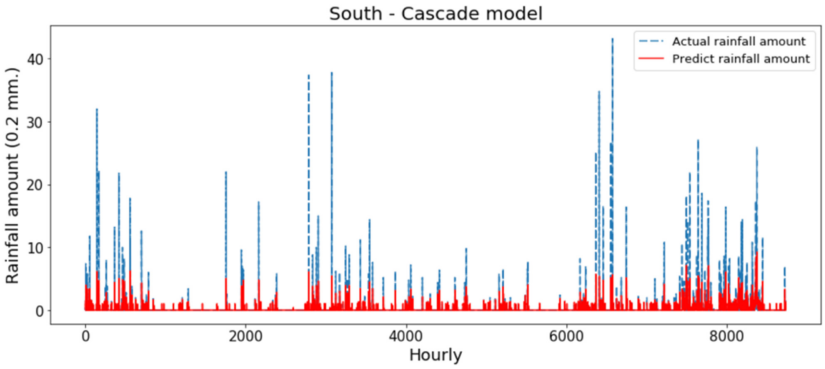

Fig.14: *Predictive rainfall using CNN-GRU for the southern region.*

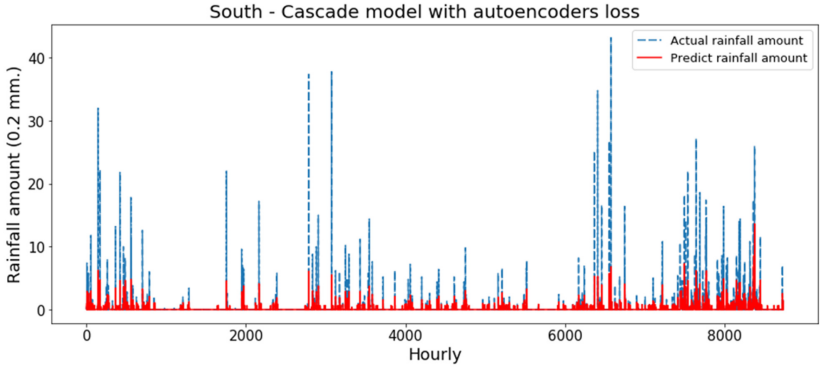

Fig.15: *Predictive rainfall using CNN-AE-GRU for the southern region.*

fall prediction using our proposed loss function called "autoencoder loss". The experiment was conducted on the rainfall dataset in Thailand for years between 2012 and 2018. For increasing the rainfall detection rate, the results show that the focal loss improves classification accuracy in all regions. This is especially noticable in the southern region, with a 59.02% F1 score-improvement. Moreover, we succeed in rainfall prediction improvement by adding autoencoder loss. Our proposed model is the best at accurately forecasting rainfall amounts in all regions, especially in the eastern region, with a 1.50% RMSE reduction.

ACKNOWLEDGMENT

The dataset used in this paper was provided by the Hydro-Informatics Institute (Public Organization), Ministry of Higher Education, Science, Research and Innovation, Thailand.

References

[1] S. Sooktawee and G. Rajchakit, "Development simulation of an unseasonal heavy rainfall event over southern Thailand by WRFROMS coupling model," *International Journal*, vol. 18, no. 65, pp. 55-63, 2020.

[2] M. Murat, I. Malinowska, M. Gos, and J. Krzyszczak, "Forecasting daily meteorological time series using ARIMA and regression models," *International agrophysics*, vol. 32, no. 2, pp. 253-264, 2018.

[3] S. Wang, J. Feng, and G. Liu, "Application of seasonal time series model in the precipitation forecast," *Mathematical and Computer Modelling*, vol. 58, no. 3-4, pp. 677-683, 2013.

[4] R. Mittelman, B. Kuipers, S. Savarese, and H. Lee, "Structured recurrent temporal restricted Boltzmann machines," in *International Conference on Machine Learning*, 2014, pp. 1647-1655.

[5] A. S. Cofino, R. Cano Trueba, C. M. Sordo, and J. M. Gutiérrez Llorente, "Bayesian networks for probabilistic weather prediction," 2002.

[6] S. Cramer, M. Kampouridis, A. A. Freitas, and A. K. Alexandridis, "An extensive evaluation of seven machine learning methods for rainfall prediction in weather derivatives," *Expert Systems with Applications*, vol. 85, pp. 169-181, 2017.

[7] N. Q. Hung, M. S. Babel, S. Weesakul, and N. Tripathi, "An artificial neural network model for rainfall forecasting in Bangkok, Thailand," *Hydrology & Earth System Sciences*, vol. 13, no. 8, 2009.

[8] J. N. Liu, Y. Hu, J. J. You, and P. W. Chan, "Deep neural network-based feature representation for weather forecasting," in *Proceedings on the International Conference on Artificial Intel-*

ligence (ICAI), 2014: The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2014.

[9] K. Boonyuen, P. Kaewprapha, and P. Srivihok, "Daily rainfall forecast model from satellite image using convolution neural network," in *2018 International Conference on Information Technology (InCIT)*, 2018: IEEE, pp. 1-7.

[10] K. Boonyuen, P. Kaewprapha, U. Weesakul, and P. Srivihok, "Convolutional neural network inception-v3: a machine learning approach for leveling short-range rainfall forecast model from satellite image," in *International Conference on Swarm Intelligence*, 2019: Springer, pp. 105-115.

[11] E. Hernández, V. Sanchez-Anguix, V. Julian, J. Palanca, and N. Duque, "Rainfall prediction: A deep learning approach," in *International Conference on Hybrid Artificial Intelligence Systems*, 2016: Springer, pp. 151-162.

[12] M. Qiu, P. Zhao, J. Huang, X. Shi, X. Wang and W. Chu, "A short-term rainfall prediction model using multi-task convolutional neural networks," in *2017 IEEE International Conference on Data Mining (ICDM)*, 2017: IEEE, pp. 395-404.

[13] F. Manokij, K. Sarinnapakorn, and P. Vateekul, "Forecasting Thailand's Precipitation with Cascading Model of CNN and GRU," in *2019 11th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2019: IEEE, pp. 1-6.

[14] W. Kliengchuay, A. Cooper Meeyai, S. Worakhunpiset, and K. Tantrakarnapa, "Relationships between meteorological parameters and particulate matter in Mae Hong Son province, Thailand," *International Journal of Environmental Research and Public Health*, vol. 15, no. 12, p. 2801, 2018.

[15] F. J. J. Joseph, "IoT Based Weather Monitoring System for Effective Analytics," *International Journal of Engineering and Advanced Technology*, vol. 8, no. 4, pp. 311-315, 2019.

[16] F. J. J. Joseph, "IOT Based Unified Approach To Predict Particulate Matter Pollution In Thailand."

[17] T. Kieu, B. Yang, C. Guo, and C. S. Jensen, "Outlier Detection for Time Series with Recurrent Autoencoder Ensembles," in *IJCAI*, 2019, pp. 2725-2732.

[18] Y. Xia, X. Cao, F. Wen, G. Hua, and J. Sun, "Learning discriminative reconstructions for unsupervised outlier removal," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1511-1519.

[19] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "LSTM-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, 1998.

[21] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[22] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980-2988.

[23] G. S. Babu, P. Zhao, and X.-L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life," in *International conference on database systems for advanced applications*, 2016: Springer, pp. 214-228.

[24] K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929-1958, 2014.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

Fuenglada Manokij received B.Eng in electronic and computer engineering from King Mongkut's University of Technology North Bangkok (KMUTNB), Thailand in 2013. She received her M.Sc. degree in computer science from Chulalongkorn university, Thailand in 2019. Currently, she works as a senior engineer at at Advanced Wireless Network (AWN). Her research involves applied deep learning techniques in meteorological field such as rainfall data.

Kanoksri Sarinnapakorn received her Ph.D. degree in Electrical and Computer Engineering from University of Miami, USA, got B.S. and M.S. degrees in Statistics from Kasetsart University, Thailand, and got another M.S. in Computer Science from Fairleigh Dickinson University, USA. Her expertise is in data science, machine learning and advanced statistical data analysis. Currently she is Head of Climate and Weather Section and Hydro Data Science Section of Hydro-Informatics Institute (Public Organization). Her work is related to weather forecast modeling and sub-seasonal to seasonal prediction of Thailand for water resource management. She also participates in data quality control and data governance working group of Hydro-Informatics Institute.

Peerapon Vateekul received his Ph.D. degree from Department of Electrical and Computer Engineering, University of Miami (UM), Coral Gables, FL, U.S.A. in 2012. Currently, he is an assistant professor at Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Thailand. Also, he is a deputy head of the department in academic affairs. His research falls in the domain of machine learning, data mining, deep learning, text mining, and big data analytics. To be more specific, his works include variants of classification (hierarchical multi-label classification), data quality management, and applied deep learning techniques in various domains, such as, medicinal images and videos, satellite images, meteorological data, and text.