Algorithm design for wind prediction in Berakit Bay, Bintan Island using Long Short-Term Memory (LSTM) method

D R Pratama, I Jaya* and M Iqbal
Department of Marine Science and Technology, Faculty of Fisheries and Marine Science, IPB University, Dramaga, Bogor 16680, West Java, Indonesia
*E-mail: indrajaya@apps.ipb.ac.id

Abstract. Wind speed is a crucial parameter alongside coastal areas, especially Indonesia. Above average wind speed can cause harmful effects on human activities. This study uses wind speed data from Berakit Bay, Bintan Island is a potential location for coastal community settlement, fisheries, and tourist activities. The wind parameter then predicted using the Long Short-Term Memory or LSTM algorithm. This algorithm is able to study long-term dependencies by converting simple nervous system designs into specialized blocks containing cells. It is suitable to be applied to long-term wind predictions where the wind speed at this time is very influential with the wind speed in the future. In preparing the LSTM, the data preprocessing and the architecture used will determine the prediction results. In this study, four different architectures were made in order to determine the most optimal architecture. The results show that the LSTM architecture is able to obtain a relatively good RMSE value of 1.87 and an accuracy of 39.40% with the use of two LSTM layers, 256 units in the first layer and 128 in the second layer. The LSTM algorithm in predicting wind can also be applied to other areas in Indonesia.

Keywords: LSTM, prediction, wind speed

1. Introduction
One of the physical parameters that cannot be separated from human life is wind. This parameter has an important role in various aspects of human life. Wind is one of the important meteorological parameters in coastal areas, especially in sea transportation [1]. Another activity on the coast that uses wind parameters as operational considerations is traditional fishing activities that still use simple fishing gear. In addition, fishing operations are also depending on wind to maneuver around the fishing ground [2]. This wind parameter can also have a negative impact that causes harm to human activities [1]. This negative impact occurs when the wind blows at a speed and strength above the normal limit, so that wind speed predictions can be applied to prevent the negative impact of the wind.

Bintan Regency is located between 1°05'03.94" North Latitude and 104°28'56.23" East Longitude. Bintan Regency is bordered by Natuna Regency in the north, Lingga Regency in the south, Tanjungpinang City and Batam City in the west, and West Kalimantan Province in the east. The land area of Bintan Regency reaches 1,320.10 km² and consists of 272 large and small islands. Bintan Regency has a tropical climate with the lowest temperature of 21.6 °C, the highest temperature of 37.7 °C, and air humidity of about 82% [3]. The wind speed in Bintan Regency is strongly influenced by the movement of the west and east monsoons where there is a movement of air mass from the northern hemisphere (BBU) to the southern hemisphere (BBS) or vice versa caused by the apparent movement of the sun [4-5].

One method that can be used to predict the wind is to use a Recurrent Neural Network (RNN) which is a neural network structure model with temporal behavior that can be used as sound frequency
recognition, language modelling to predict a parameter by studying input data and match it with the results or outputs [6-8]. RNN is a part of deep learning and can be implemented using the python programming language [9]. The RNN algorithm itself is not able to connect the information carried by the two data inputs if they are too far apart in the input sequence or sequence [8]. This causes the loss of the predictive gradient, so the processed numbers cannot be said to be predictive. The architecture used to maintain the gradient is to use a long short-term memory or LSTM architecture. LSTM-RNN is able to study long-term dependencies by converting simple nervous system designs into specialized blocks containing cells. These cells contain gates or gates. Gates has a different function in each cell, namely to store relevant data (input), forget irrelevant data in the sequence (forget), and output. The characteristics of the LSTM-RNN are suitable to be applied to the prediction of wind direction and speed on the coast, where the long-term context must be considered [8, 10].

Research on wind prediction using the LSTM-RNN algorithm has previously been carried out which resulted in an RMSE value of 6888.37 [11]. Better results using the same algorithm were also used by [12] with an RMSE value of 0.105. [1] have conducted a similar study using two different algorithms, namely Adaptive Neuro Fuzzy (ANFIS) and Radial Basis Function (RBF). Both algorithms produce a fairly good RMSE value with RMSE values of 1.14 and 0.17, respectively. The algorithm that will be used (LSTM-RNN) in predicting wind parameters on Bintan Island is expected to be able to compete with previous research.

2. Materials and Methods

The proposed approach for *Enhalus acoroides* detection and segmentation is performed in five steps: data collection, color correction, data labelling, training, and finally, evaluation of the model. All these steps illustrated in Figure 1 are discussed as follows.

2.1. Data Collection

The materials used in this study include data on U components and V components (east-west components and north-south components) obtained from the 5th generation ECMWF (ERA5). The data obtained from ECMWF is the result of global observations, which are then combined into a dataset with many choices of physical parameters that can be downloaded by the user. Parameters, sources, resolutions, and data formats can be seen in Table 1.

| Parameter                  | Source                                                     | Resolution        | Format   |
|---------------------------|------------------------------------------------------------|-------------------|----------|
| 10 meters U component     | ERA5 Hourly Data on Single Levels from 1979 to Present     | 0.5° lat x 0.5° lon | netCDF (.nc) |
|                           | https://cds.climate.copernicus.eu/                          |                   |          |
| 10 meters V component     | ERA5 Hourly Data on Single Levels from 1979 to Present     | 0.5° lat x 0.5° lon | netCDF (.nc) |
|                           | https://cds.climate.copernicus.eu/                          |                   |          |

2.2. U and V Wind Velocity Component Conversion

Conversion is carried out using Python programming language to get wind speed parameters. The U component is the value of the wind speed in the x-direction or in the East-West direction or what is called the zonal wind, while V is the wind speed value in the y or North-South vector direction, which is called the meridional. The resultant wind speed value is obtained from the first equation below where C is the wind speed in m/s.

\[ C = \sqrt{(U)^2 + (V)^2} \] (1)

Note:

- \( C \) = wind speed (m/s)
- \( U \) = zonal component
- \( V \) = meridional component

The converted data is then combined using the Pandas library to obtain a two-dimensional time series data frame table where the first column is time and the second column is for wind speed. The data frame
is then exported using csv format or comma-separated values. This format is used for the purpose of inputting data at a later stage in Google Collaboratory.

2.3. Data Preprocessing

Data preprocessing is a process where the data to be used as training is prepared in advance. In this study, data preprocessing was prepared by going through a process to handle missing or empty data in a way such as finding the average attribute for the same class [13]. LSTM is sensitive to data scale therefore data needs to be normalized using MinMaxScaler with a range (-1, 1) [14]. The MinMax method is a method of normalizing a value by performing a linear transformation of the original data. The normalization uses the following equation:

\[ X' = \frac{X - \min X}{\max X - \min X} \]  

Note:
- \( X' \) = Normalization value
- \( X \) = Data
- \( \min X \) = minimum value
- \( \max X \) = maximum value

2.4. LSTM

Long Short-Term Memory or abbreviated as LSTM is one type of RNN architecture which was first coined by Hochreiter and Schmidhuber in 1997. LSTM and RNN have a difference where LSTM able to learn information by sorting out information that must be discarded and information that must be kept in memory cells. This LSTM ability is obtained from the LSTM neural structure which is different from ordinary RNN nerves. Each LSTM neuron has a memory cell and gate units. Memory cells function to store information, while gate units have the function of managing stored information. The following is how each process works on LSTM neurons [6, 12].

The first process as can be seen in Figure 1 is filtering the information that must be discarded. Forget gate, which is a sigmoid function and is denoted by \( f_t \), will read the values of \( h_{t-1} \) and \( X_t \), returning a value of 0 or 1 for each element in \( C_{t-1} \). A value of 0 indicates that the element (information) should be discarded. On the other hand, a value of 1 indicates that the element (information) should be retained [6, 12].

**Figure 1.** Forget gate.

\[ f_t = \sigma(W_f[h_{t-1}, X_t] + b_f) \]  

Note:
- \( f_t \) = Forget gate
- \( \sigma \) = Sigmoid function
- \( W_f \) = Weight value on forget gate
- \( h_{t-1} \) = Output value from previous cell
- \( X_t \) = Input value at t time
- \( b_f \) = Bias value on forget gate

In Figure 2 which is the next process, there are input gates and tanh. The input gate is also a sigmoid function. The input gate or \( i_t \) determines which information will be updated and the tanh function will create a new vector, namely \( \tilde{C}_t \) [6, 12].
\begin{equation}
i_t = \sigma(W_i[h_{t-1}, X_t] + b_c)
\end{equation}

Note:
\begin{itemize}
  \item $i_t$ = Input gate
  \item $\sigma$ = Sigmoid function
  \item $W_i$ = Weight value at input gate
  \item $h_{t-1}$ = Output value from previous cell
  \item $X_t$ = Input value at t time
  \item $b_f$ = Bias value on input gate
\end{itemize}

\begin{equation}
C_t = tanh(W_c[h_{t-1}, X_t] + b_c)
\end{equation}

Note:
\begin{itemize}
  \item $C_t$ = Cell state (memory cell)
  \item $tanh$ = Tanh sigmoid function
  \item $W_c$ = Weight value at memory cell
  \item $h_{t-1}$ = Output value from previous cell
  \item $X_t$ = Input value at t time
  \item $b_c$ = Bias value on memory cell
\end{itemize}

The next process that can be seen in Figure 3 is updating the memory cell from $C_{t-1}$ to $C_t$. The value of $C_t$ is obtained from multiplying $f_t$ with $C_{t-1}$ plus multiplying $i_t$ with $C_t$. [6, 12].

\begin{equation}
C_t = f_t(C_{t-1} + i_t[C_t])
\end{equation}

Note:
\begin{itemize}
  \item $C_t$ = Cell state (memory cell)
  \item $f_t$ = Forget gate
  \item $i_t$ = Input gate
\end{itemize}

The last process in Figure 4 is to determine the output. The sigmoid function (output gate, $o_t$) will be executed in this process to get the parts of the memory cell that will be output and tanh will process $C_t$ to get a value between 1 and -1. This is the main reason why the data is normalized at the preprocessing stage [6, 12].
Figure 4. Output gate.

\[ C_t = f_t( [C_{t-1}] + i_t[C_t] ) \]  

(7)

Note:
- \( o_t \) = Output gate
- \( \sigma \) = Sigmoid function
- \( W_o \) = Weight at output gate
- \( h_{t-1} \) = Output value from previous cell
- \( X_t \) = Input value at t time
- \( b_o \) = Bias value on memory cell

\[ h_t = o_t [ \tanh (C_t) ] \]  

(8)

Note:
- \( h_t \) = Final result
- \( o_t \) = Output gate value
- \( tanh \) = Tanh sigmoid function
- \( C_t \) = Cell state (memory cell)

2.5. Model Testing

The accuracy of the prediction results is calculated using the Root Mean Squared Error equation or abbreviated as RMSE. [15] stated that the RMSE calculation is intended to see how much error is generated from a prediction model made so that the closer the error value is to 0, the better the model made will be. The RMSE equation was chosen because it avoids the use of absolute values, which is highly undesirable in many mathematical calculations [16]. The following is the RMSE equation.

\[ RMSE = \sqrt{ \frac{\sum (y_i - y_0)^2}{N} } \]  

(9)

Note:
- \( y_i \) = Prediction results
- \( y_0 \) = Real data
- \( N \) = Number of samples

The correlation test is carried out to measure the strength of the relationship between two such variables through a number called the correlation coefficient. The correlation coefficient as a measure of the linear relationship between the random variables X and Y is denoted by r. The equation used is the Pearson correlation coefficient. This equation is commonly used in calculating the linear correlation between two variables [17] and is commonly used in calculating the accuracy of artificial or artificial neural network models [18]. The following is the correlation coefficient equation and the linear regression equation used.

\[ y = ax + b \]  

(10)

information:
- \( x \) = Value on horizontal axis (real data)
- \( y \) = Value on vertical axis (prediction result)
3. Results and Discussion

3.1. Dataset Characteristics

The dataset that will be used in the prediction is time data that acts as an index and wind speed. Prior to preprocessing, it is necessary to ensure that the dataset to be used is continuous by looking at the characteristics of the data as presented in Table 2.

| Parameters       | Value                                      |
|------------------|--------------------------------------------|
| Wind speed (m/s) | Count: 56976                                 |
|                  | Mean: 4.153                                  |
|                  | Std: 1.90                                    |
|                  | Min: 0.048                                   |
|                  | Max: 15.160                                  |
| Time             | Range: 6 hours                               |
|                  | Timespan: 1981-01-01 06:00:00 – 2020-01-01 00:00:00 |
| Coordinate       | Longitude: 104.58                            |
|                  | Latitude 1.19                                |

The amount of data used as a training dataset is 56,976 data which has a standard deviation of 1.90 m/s, an average speed of 4.153 m/s with a minimum speed of 0.048 m/s, and a maximum of 15.160 m/s from 1981 to 2019. Total the data used determines the prediction pattern that is formed on the prediction results. This is in accordance with the research of [19] which uses time series data with a long period of time to determine the prediction pattern formed on a model and according to [20], the amount of past data (historical data) is able to fix the problem of gradient loss in the prediction of a parameter. The following is a visualization of wind speed data that is processed using the matplotlib library. The downloaded research data set has a time span from January 1, 1981 at 6 am to January 1, 2021 at 12 pm with a distance of data retrieval every 6 hours per day so that there are 4 data per day at 6 am, 12 noon, 6 pm, and 12 am. The visualization of the wind speed dataset can be seen in Figure 5.

![Figure 5. Wind speed dataset visualization.](image)

3.2. Preprocessing

The main preprocessing is done by changing the wind speed data scale into a range (-1.1). Scale changes (rescaling) are carried out because LSTM cells use the sigmoid function, so they are sensitive to changes in relatively large numbers and rescaling needs to be done to avoid problems in the learning process (training) [14, 21, 22]. Rescaling the data is done by applying the normalization equation using MinMaxScaler on the dataset. The following as presented in Figures 6a and 6b is a graphical visualization of the comparison of datasets before normalization and after normalization. Visually the graphic pattern does not change, but changes occur in the numbers where after normalization, the minimum value is -1 and the maximum value is 1.
Hyperparameters

The training hyperparameters used include Epochs, Batch Size, and Learning Rate. Each parameter has its own function. Epochs is a condition when the dataset used goes through one LSTM network process back and forth (forward and backward), while Batch Size is the number of training datasets in one batch, batch and batch size are two different things where we cannot include the entire dataset in one LSTM network so that it is divided into the number of batches and the total training dataset is determined in the number of batch sizes used. Another hyper-parameter is the learning rate which is a parameter that can be configured to have a very small positive value of 0.0 to 1.0. This parameter is used to determine the size of the step that determines whether the training results are getting better (the loss value decreases) or worsens (the loss value increases) [23], so that this becomes a reference in using the minimum learning rate with a value of 0.01 for all architecture. LSTM hyperparameters are accompanied by callbacks that have the function of storing weight values during the training process. Callbacks are used to avoid overfitting or underfitting events in the training process [24] so that 3 types of callbacks are used, namely CSV Logger which stores weight values in CSV format, Checkpoint Model which secures weight values in h5 format, and Tensorboard which is a separate tool to display a graph of training results.

| Hyperparameter   | Value |
|------------------|-------|
| Epochs           | 200   |
| Batch size       | 1024  |
| Learning rate    | 0.01  |

Algorithm Preparation

The use of the number of LSTM units can be seen in Table 4. There are 4 types of LSTM architectures which are distinguished from the number of LSTM Units. The first and second architectures use 2
layers, while the third and fourth architectures only use 1 layer. The difference in the use of layers and LSTM Units aims to see the difference in the predicted values produced, as described in [25] that the number of LSTM networks and layers used in a model determines the model in making decisions about an outcome, so this becomes a reference in determining the most optimal architecture used to predict wind. [9] explains that the number of units used affects the improvement of the prediction results, so that the number of multiples of 8 is taken wherein layer 1 uses multiples of 8 which are larger than layer 2 to avoid errors in the model when making decisions on results.

| Model       | LSTM Units Layer 1 | LSTM Units Layer 2 |
|-------------|--------------------|--------------------|
| Architecture 1 | 256                | 128                |
| Architecture 2 | 128                | 64                 |
| Architecture 3 | 256                |                    |
| Architecture 4 | 128                |                    |

### 3.5. Prediction Results

Predictions were made throughout 2020 at the research location. Figure 7 shows the comparison between the training dataset (blue), real data (purple), and predicted data (red). Visually, the prediction results can follow the same trendline pattern as the original data. This shows that the prediction results are able to follow the pattern formed from the training dataset and learn the patterns of events with respect to time. Having a pattern that is similar, but not exactly the same, the difference between the predicted results and the real data is caused by a factor where according to a predictive model the value is obtained by studying the values of past events [6, 26], while values in real data are obtained from events that occur at a location. The comparison between the predicted results, real data, and data quoted from [4] is presented as follows.

![Figure 7](image1.png)

**Figure 7.** Comparison between real (violet) and predicted (red) value. The blue color is the train data. X-axis represent time and Y-axis represent wind speed.
Table 5. Average comparison between predicted value, real data, and Bintan statistics.

| Average   | Prediction (m/s) | Real Data (m/s) | Bintan statistics (m/s) |
|-----------|-----------------|-----------------|-------------------------|
| Annual    | 4.43            | 4.99            | 3                       |
| Q1        | 5.81            | 5.96            | 3.67                    |
| Q2        | 4.19            | 3.44            | 2.67                    |
| Q3        | 5.36            | 4.01            | 3                       |
| Q4        | 4.59            | 4.31            | 2.67                    |

Based on Table 5, the average taking is divided into 5 parts, namely annual, quartile 1 (January, February, March), quartile 2 (April, May, June), quartile 3 (July, August, September), and quartile 4 (October, November, December). In the first quartile, the prediction data produces a value of 5.81 m/s and the real data is 5.96 m/s. The large differences were found in the second and third quartile, relatively. The second quartile of the predicted results shows 4.19 m/s and the real data is 3.44 m/s, while in the third quartile the predicted data shows 5.34 m/s and the real data is 4.01 m/s. The fourth quartile does not show much difference with the predicted results of 4.59 m/s and the real data of 4.31 m/s.

3.6. Model Accuracy Test
Table 6 shows the RMSE values of the four architectures created. The smallest RMSE value was obtained by Architecture 1 with a value of 1.87, and the largest by Architecture 4 with a value of 1.93, while Architecture 2 and 3 got the same value, namely 1.92. The use of RMSE can determine the best model improvement because it displays the actual error [16], therefore it can be said that the most optimal architectural model is Architecture 1 using 2 layers of LSTM, while the less optimal model is Architecture 4. Results The RMSE of the four architectures is able to match the results of previous studies by [11] using the same algorithm as the results of RMSE 6888.37, but have not received a value smaller than the research of [12] who got an RMSE value of 0.105 with the same algorithm and [26] with an RMSE value of 0.09 using the ANN algorithm.

Table 6. RMSE Value.

| Model      | RMSE  |
|------------|-------|
| Architecture 1 | 1.87  |
| Architecture 2 | 1.92  |
| Architecture 3 | 1.92  |
| Architecture 4 | 1.93  |

Figure 8. Correlation between Real Data (x-axis) and Predicted Value (y-axis).

The accuracy of the prediction results is visualized using linear regression which is presented in Figure 19 below where the x-axis is the real data and the y-axis is the predicted result. Based on the graph, the correlation coefficient between real data and prediction results is 0.3940 or 39.4%. This
indicates that there is a small relationship between the predicted results and the real data. The value of the correlation coefficient is relatively small due to the large amount of training data as described by [27] that the greater the amount of training data, the smaller the accuracy produced, but the patterns formed will be relatively similar.

4. Conclusion
The LSTM algorithm is able to predict wind parameters on the coast of Teluk Berakit, Bintan Island. Based on the training results, the most optimal architecture is Architecture 1 which uses 2 LSTM layers with an RMSE value of 1.87, while the less optimal architecture is Architecture 4 which uses only 1 LSTM layer with an RMSE value of 1.93.

References
[1] Nikentari N, Bettiza M and Pratiwi H S 2018 Prediksi kecepatan angin menggunakan Adaptive Neuro Fuzzy (ANFIS) dan Radial Basis Function Neural Network (RBFNN) JEPIN 4(1) 70-75
[2] Uifa M 2018 Persepsi masyarakat nelayan dalam menghadapi perubahan iklim (ditinjau dalam aspek sosial ekonomi) Journal Pendidikan Geografi 23(1) 41-49
[3] Badan Pusat Statistik Kabupaten Bintan 2021 Kabupaten Bintan dalam Angka Edisi 2021 (Bintan: BPS Kabupaten Bintan)
[4] Dida H P, Suparman S and Widhiyanuriyawan D 2016 Pemetaan potensi energi angin di perairan Indonesia berdasarkan data satelit QuickSCAT dan WindSat Jurnal Rekayasa Mesin, 7(2) 5-101
[5] Negara G S 2020 Dampak lingkungan terhadap pencemaran laut di pesisir utara Pulau Bintan selama musim angin utara Jurnal Saintek Maritime 20(2) 137-144
[6] Cheng L, Zang H, Ding T, Sun R, Wang M, Wei Z and Sun G 2018 Ensemble recurrent neural network based probabilistic wind speed forecasting approach Energies 11(8) 1958-1980
[7] Pradhan P P and Subudhi B 2019 Wind speed forecasting based on wavelet transformation and recurrent neural network Int. J. Numer. Model. 33(1) 1-11
[8] Wu W, Chen K, Qiao Y and Lu Z 2016 Probabilistic short-term wind power forecasting based on deep neural networks Proc. Int. Conf. on Probabilistic Methods Applied to Power Systems (Beijing, 16-20 October 2016) (Beijing: IEEE Publications) pp 1-8
[9] Chollet F 2018 Deep Learning with Python (New York: Manning Publications) 361 p
[10] Abdel-Nasser M and Mahmoud K 2019 Accurate photovoltaic power forecasting models using deep LSTM-RNN Neural. Comput. Appl. 31(7) 2727-2740
[11] Rizal A A and Soraya S 2018 Multi time steps prediction dengan recurrent neural network long short-term memory MATERIK 18(1) 115-124
[12] Andiyantama M Q, Zahira I and Irawan A 2021 Prediksi energi listrik kincir angin berdasarkan data kecepatan angin menggunakan LSTM JITCE. 5(1) 1-7
[13] Rizki M, Basuki S and Azhar Y 2020 Implementasi deep learning menggunakan arsitektur long short-term memory untuk prediksi curah hujan kota Malang REPOSITOR. 2(3) 331-338
[14] Delgado I and Fahim M 2021 Wind turbine data analysis and LSTM-based prediction in SCADA system Energies 14(1) 125-140
[15] Wang X, Guo P and Huang X 2011 A review of wind power forecasting models Energy Procedia 12 770-778
[16] Chai T and Draxler R R 2014 Root mean square error (RMSE) or mean absolute error (MAE) arguments against avoiding RMSE in the literature Geosci. Model Dev. 7 1247-1250
[17] Walpole R E 2005 Pengantar Statistika (Jakarta: Gramedia Pustaka Utama)
[18] Lu W, Li J, Li Y, Sun A and Wang J 2020 A CNN-LSTM-based model to forecast stock prices Complexity 2020 1-10
[19] Salman A G, Heryadi Y, Abdurahman E and Suparta W 2018 Single layer and multi-layer long short-term memory (LSTM) model with intermediate variables for weather forecasting Procedia Comput. Sci. 135 89-98
[20] Samarasinghe S 2007 Neural Networks for Applied Sciences and Engineering from Fundamentals to Complex Pattern Recognition (Florida: Auerbach Publications)
[21] Zaytar M A and Amrani C E 2016 Sequence to sequence weather forecasting with long short-term memory recurrent neural networks Int. J. Comput. Appl. 143(11) 7-11
[22] Salman A G, Heryadi Y, Abdurahman E and Suparta W 2018 Weather forecasting using merged long short-term memory model Bull. Electr. Eng. Inform. 7(3) 377-385
[23] Goodfellow I, Bengio Y and Courville A 2016 Deep Learning (Massachusetts: MIT Press) 789 p
[24] Wang F, Decker J, Wu X, Essertel G and Rompf T 2018 Backpropagation with continuation callbacks: foundations for efficient and expressive differentiable programming Adv. Neural Inf. Process. Syst. 10180-10191
[25] Zhao Z, Chen W, Wu X, Chen P C Y and Liu J 2017 LSTM network: a deep learning approach for short-term traffic forecast IET Image Process. 11(1) 68-75
[26] Syukri and Samsuddin 2018 Pengujian artificial neural network (ANN) untuk prediksi kecepatan angin Jurnal Nasional Komputasi dan Teknologi Informasi. 2(1) 43-47
[27] Zaunseder E, Muller L and Blankenburg S 2018 High accuracy forecasting with limited input data: using FFNNs to predict offshore power generation Proc. Int. Symp. on The Ninth International Symposium on Information and Comunications Technology (Danang City Vietnam, 6-7 December 2018) (New York: Association for Computing Machinery) pp 61-68