2020 3rd International Conference on Information and Communications Technology (ICOIACT) | 978-1-7281-7356-6/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICOIACT50329.2020.9332074

High Accuracy Conversational AI Chatbot Using Deep Recurrent Neural Networks Based on BiLSTM Model

Prasnurzaki Anki
Department of Mathematics
Universitas Indonesia
Depok, Indonesia
prasnurzaki.anki@sci.ui.ac.id

Alhadi Bustamam
Departement of Mathematics
Universitas Indonesia
Depok, Indonesia
alhadi@sci.ui.ac.id

Herley Shaori Al-Ash
Departement of Mathematics
Universitas Indonesia
Depok, Indonesia
herley.shaori@sci.ui.ac.id

Devvi Sarwinda
Departement of Mathematics
Universitas Indonesia
Depok, Indonesia
devvi@sci.ui.ac.id

***Abstract*— In the modern world, chatbot programs are implementations that can be used to store data collected through a question and answer system and then can be applied in the Python program to optimize the results based on highly rated questions asked in a service center. The application of chatbots in the Python program can use various models. Specifically in this program, the BiLSTM model will be applied. The output produced from the chatbot program with the application of the BiLSTM model is in the form of accuracy and also data set that matches the information the program user enters in the chatbot's input dialog box. The selection of models that can be applied to the program is based on data which can affect program performance, with the objective of the program which can determine the high or low level of accuracy that will be generated from the results obtained through a program, which can be a major factor in deciding the selected model. Based on the various considerations that are the requirements for choosing a model of a program, in the end the BiLSTM model is selected will be applied to the program. In addition to model selection, the next step is to determine the method used in the program, in this program the greedy method is a form of implementation of the BiLSTM model with the aim that when running the program, data processing time can be faster, and increase the value of the model selected in program. In addition, supporting attributes such as the seq2seq model are a determining factor in a program that can function to verify whether data processing matches the criteria that can be used as new in data processing. In addition, a program evaluation method is needed that can be used to verify whether the program output matches the data expected by the user. Based on the application of the BiLSTM model into the chatbot, it can be concluded that with all program test results consisting of a variety of different parameter pairs, it is stated that Parameter Pair 1 (size_layer 512, num_layers 2, embedded_size 256, learning_rate 0.001, batch_size 32, epoch 20) from File 3 is the BiLSTM Chatbot with the avg accuracy value of 0.995217 which uses the BiLSTM model is the best parameter pair.**

Keywords— chatbot, program, BiLSTM, accuracy, input, output

I. INTRODUCTION

In the modern world, there are numerous questions being submitted by customers to a customer service system and many possibilities of answers that cannot answered in quickly. After considering time limitations, reducing waiting time, and increasing service results, there needs to be programmed to answer those problems. Choosing chatbots as a solution for answering questions based on various problems that consumers face can help them obtain answers quickly. Arranging the structure of the display components of the chatbot in the Python program is easier to use and is also more productive in interpretation programs [1]. Based on a set of sequences, computational methods must solve two major problems: effectively representing a sequence as a feature vector that can be analyzed, and designing a model that can identify data quickly and accurately [2].

Furthermore, after knowing the various things needed to create a chatbot, keep in mind what to expect once the chatbot is created. After the chatbot program is created, it is hoped that the input data generated through the question and answer system will be prepared and then implemented into the Python program, so that in the end consumers can get answers to their questions that were raised to the chatbot. A chatbot is a system that receives user input with continuous responses. Parts of the chatbot can be built from an encoder-decoder architecture [3]. Simply put, a chatbot is a simple robot form in the shape of a program that answers questions from users which produces output data in the form of answers.

A chatbot is a software tool that interacts with users on a certain topic or in a specific domain in a natural, conversational way using text and voice. For many different purposes, chatbots have been used across a wide range of domains, including marketing, customer service, technical support, as well as education and training [4]. In this study, the chatbot that

will be created will be limited in handling questions that will be asked by consumers based on a predetermined dataset. In the build of chatbot, it takes steps that contain materials and methods that will make the chatbot have high accuracy in meeting user needs, the materials and methods to be used will be discussed in section II.

II. MATERIALS AND METHOD

There are set of sentences from datasets that can used to build a chatbot program based on the BiLSTM model, multiple parameter pairs, and the Greedy method. Inputs from the user can be commands to run the chatbot program with results in the form of sentences which contain information according to the initial input that the user enters.

A. Steps in making a chatbot

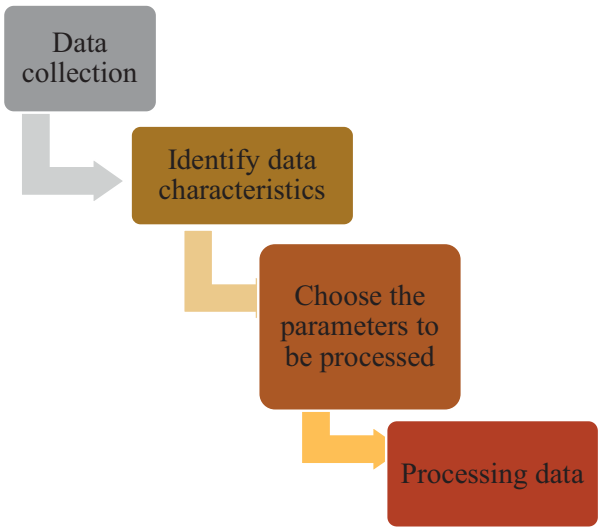

Fig. 1. Data management diagram

There are 4 steps to creating a chatbot, namely: data identity, data input for the question and answer system, compiling a chatbot program, and evaluating the output.

- Data identity

  The data to be used in this program is the Cornell Movie Dialog Corpus which is a dataset containing a corpus which contains a large collection of metadata-rich fictional conversations extracted from film scripts, which have 220,579 conversation exchanges between 10,292 pairs of movie characters, involving 9,035 characters from 617 films, and a total of 304,713 sayings [5]. The data used is the 2018 data. Based on the parameter selection stage, the conversation data will then be selected and then processed from the input questions that come from the dataset, which will then be answered by the machine as a form of response to the chatbot program. The answers to the questions will produce a response for individuals who want to obtain answers about matters related to the characteristics of film data. To manage this data, below is a data management diagram that explains the steps from collecting to processing the data.

- Question and answer system input data

  In carrying out input from program users, files containing dialog sentences in the film are inputted into the program as user input. Then, the input will be processed to obtain the output of the program in the form of a dialogue sentence in the film, by having a relationship between the dialog in the input to the dialog in the output.

- Chatbot program development

  The following are the things that need to be considered in the preparation of the chatbot program: there will be various translation choices in the form of sequence-to-sequence, and the selection of the BiLSTM model so that we can determine the best accuracy from the response of the chatbot compared to the live human response.

- Output evaluation

The last step is evaluating whether or not the model has provided accurate results. For example, there may be a relationship between the input dialog and the output dialog that turns out to be much more accurate than an input dialog selection that the program user previously entered.

B. BiLSTM model

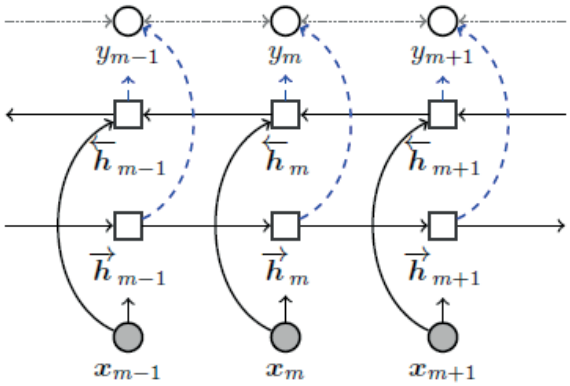

Fig. 2. BiLSTM architecture [3]

The BiLSTM model is a model that combine the advantages of the BiRNN model and the LSTM model [6]. The BiLSTM model is used to propagate the use of reverse direction as well as forward direction. The BiLSTM model is a two-way network that is used to store future data and past data which is more effective in the LSTM model [7]. Solid lines in the figure above represent calculations, while dotted lines show probabilistic dependencies and can also show optional additional probabilistic dependencies between labels in association for a combination of aspects of the BiLSTM and conditional random plane model. In the feature-based model, traits related to knowledge of shape are processed by feature suffixes in the neural network. Embeddings are a technique used to handle the sparse matrix of the bag of words.

One application of the feature of suffixes in neural networks is that they can be inserted by constructing the invisible embeddings of words from their spelling or

morphology. One way to do this is to incorporate additional two-way RNN layers, one of which is for each word in the vocabulary. The BiLSTM model is one of many parallel computing models. Based on reference from [8], parallel computing can be used to process program instructions by distributing them into multiple processors in order to reduce the running time of the program.

The first step is to encode $w^{(p)}$ dan the $w^{(q)}$ query using two LSTMs. This is known as Bidirectional LSTM (BiLSTM)

$$h^{(q)} = BiLSTM(w^{(q)};\ \theta^{(q)}) \quad (1)$$

$$h^{(p)} = BiLSTM(w^{(p)};\ \theta^{(p)}) \quad (2)$$

The questions are represented by the vector *u,* vertically combining final states from left to right and are represented by matching the ending state vertically from left to right

$$u = \left[\overrightarrow{h^{(q)}}_{M_{(q)}}; \overleftarrow{h^{(q)}}_{0}\right] \quad (3)$$

Vector $\left(u^{(q)}\right)^T$ is the result of applying the vector u with equation $h_{\boldsymbol{m}} = g(x_m, h_{m-1}), m = 1, 2, \ldots M$, (based on [3]) which has been transposed, W_a is a weight matrix with index α, $h_m^{(p)}$ is the result of implementing the hidden state with equation (1)), and vector $\widetilde{(\alpha_m)}$ is a representation of what is expected and is calculated by the following equation,

$$\widetilde{\alpha_m} = \left(u^{(q)}\right)^T W_a h_m^{(p)} \quad (4)$$

$$\alpha = \text{SoftMax}(\widetilde{\alpha}) \quad (5)$$

$$o = \sum_{m=1}^{M} \alpha_m h_m^{(p)} \quad (6)$$

In equation (4), the vector α is the result of the SoftMax function of $\widetilde{\alpha}$. In equation (5), these vectors can be arranged equal to the corresponding element in $h^{(p)}$ assuming that the candidate's answer (vector *o*) is the span of the original text.

The score of each candidate for answer *a* is calculated by the product in,

$$\hat{c} = \underset{c}{\arg\max}\, o.x_c \quad (7)$$

C. Greedy method

Besides choosing the model, the next step is to determine the method used in the program. In this program, the greedy method is chosen as a form of implementing the BiLSTM model so that in running the program, the data processing time can be faster and also helps increase the accuracy of the selected model [9].

The following is a theory regarding the greedy decoding algorithm, which is obtained based from [9]. The majority of sequences, which contain various tags, are calculated by combining the features of the input word (w_i), linkages in the *l*-th word order (w_{i-l}^{i+l}), and the order of the tags to k (t_{i-k}^{i-1}), resulting in the following equation (using θ to refer to a weighted feature instead of w to avoid confusion with the meaning of the word *w*):

$$\hat{T} = \underset{T}{argmax}\, P(T|W)$$

$$= \underset{T}{argmax} \prod_i P(t_i | w_{i-l}^{i+l}, t_{i-k}^{i-1})$$

$$= \underset{T}{argmax} \prod_i \frac{exp\ (\sum_j \theta_j f_j(t_i, w_{i-l}^{i+l}, t_{i-k}^{i-1}))}{\sum_{t' \epsilon tagset} \exp\ (\sum_j \theta_j f_j(t', w_{i-l}^{i+l}, t_{i-k}^{i-1}))} \quad (8)$$

A simple way to decode the optimal tag of a sequence $\hat{T}$ is by converting logistic regression to a sequence model is to build a local classifier that classifies each word from left to right. Using these local classifiers, a difficult classification is made for the first word in the sentence, followed by making a difficult decision for the second word, and so on. This method is called the greedy decoding algorithm, because it chooses the best tag for each word with the greedy principle, as shown in the following image:

function GREEDY SEQUENCE DECODING(words W, model P)

returns tag sequence T

for *i* = 1 **to** *length(*W*)*

$\hat{t}_i = \underset{t' \epsilon T}{\text{argmax}} P(t' | w_{i-l}^{i+l}, w_{i-l}^{i+l})$

Fig. 3. Greedy Decoding Algorithm [9]

D. Seq2seq model

The Jupyter Notebook software based on Python is chosen as the software used so that program users can see the input and output clearly from the program being run. When implementing a question and answer system into the program, it is necessary to have a seq2seq model which functions to produce various responses to the user input [9].

In section III, we will discuss how to make data implementation in the python program, which aims to be able to apply the basics of theory into the program.

III. DATA IMPLEMENTATION IN THE PYTHON PROGRAM

To implement the data into the Python program with the Jupyter Notebook software, a program planning plan will be systematically compiled as follows:

a. Choosing the software to be used

Choosing a model that is in accordance with the characteristics of the data can affect program performance, so

choosing a model that can determine whether the accuracy generated from a program is high or low is a major factor in determining the model. After considering the model selection requirements, the BiLSTM model was selected as the model to be applied into the program. In addition, supporting attributes such as the seq2seq model, are the next determining factor that can verify whether the data processing matches the criteria that is considered as the guide. In application to the program, the seq2seq model processes input sentences which will then be processed by other models and structures in the program, so that in the end it can issue various output sentences as a response generated from the chatbot program.

b. Selecting supporting models and attributes

There are various program evaluation methods such as loss, accuracy, val_loss, and val_accuracy. To introduce a method for evaluating text classification, some simple binary detection tasks will first be considered. Based on the definition from [10], accuracy is an attribute for an output value, which is described by the deviation of Δp_c from the associated input value after mathematical operations. The information content of a numeric entity can be accurately observed, if it matches the given reference. Reference is the result of a derivative of the mathematics or physical system under consideration. The accuracy value represents the measurement of the deviation from the reference.

Suppose p_0 be the numeric value in question and p_r the numeric value in reference.

For p_0 to be accurate, under the following conditions:

$$|p_r - p_0| \leq \Delta p_c \quad (9)$$

then must be satisfied in the above context, $p_r = p$.

c. Determining the program evaluation method

The accuracy of an operation is determined by how the operation maintains information from a content based on numeric values or not. Based on the definition from [10], an operation is said to be accurate if it maintains the input value of the operation as a whole. An operation will be said to be less accurate if the information can change. An operation is said to be inaccurate if the information from the content is completely lost. Based on the theory of [11], from the various loss functions, the basis of the problem will determine the selection of the loss function. In this discussion, the Sum-of-Squares Error will be used as a loss function, as follows:

$$Sum - of - Squares\ Error = \sum_{i=1}^{n}(y - \hat{y})^2 \quad (10)$$

The form of the Sum-of-Squares Error can be simplified to the sum of the difference between the predicted value and the actual value. The difference will be squared to measure the absolute value of said difference. The aim of the training is to find the best set of weights and biases to minimize loss function. In practice, to measure the error of the prediction (loss), it is necessary to multiply the error behind, and to update the values of weights and biases.

Based on the definition from [12], Validation accuracy at an early epoch (Val Acc) is the correspondence leading to an initial stop in training, where the user assumes the validation accuracy of an architecture when the initial epoch $t = T < T_{end}$ is a good predictor of the final test performance when epoch $t = T_{end}$. Validation accuracy can develop, but validation losses can stop at relatively high levels or even start to rise because the neural network can make more accurate classifications at validation points (which depend on the argmax of the logits) against the epoch set for the training data.

In section IV, a study will be conducted on applying data implementation in the python program, which will explain how the data will be processed to produce output from the data processing.

IV. APPLYING DATA IMPLEMENTATION IN THE PYTHON PROGRAM

The following discussion will describe the problem which can be used as a study of the need for implementing chatbots in the question and answer system.

A. Description of the problem

In facing the dynamics of modern times, there will be numerous questions that consumers issue based on things related to a data. Chatbots are a form of implementation of the question and answer system data. A few reasons on why it is necessary to create a chatbot program include speed comparison between manual question and answer system services carried out by humans with responses based on the chatbot program (which is the implementation of question and answer data between humans and machines) as well as easy access that can operate according to the time the program user needs.

Before discussing the components needed to make a chatbot, it is necessary to know the things that must be prepared to obtain the components needed for making the program as shown in the diagram below:

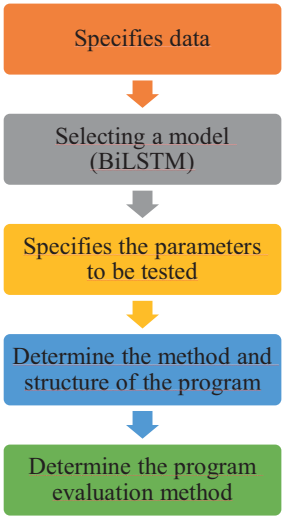

Fig. 4. Preparation diagram of the components for the chatbot program

B. Program making

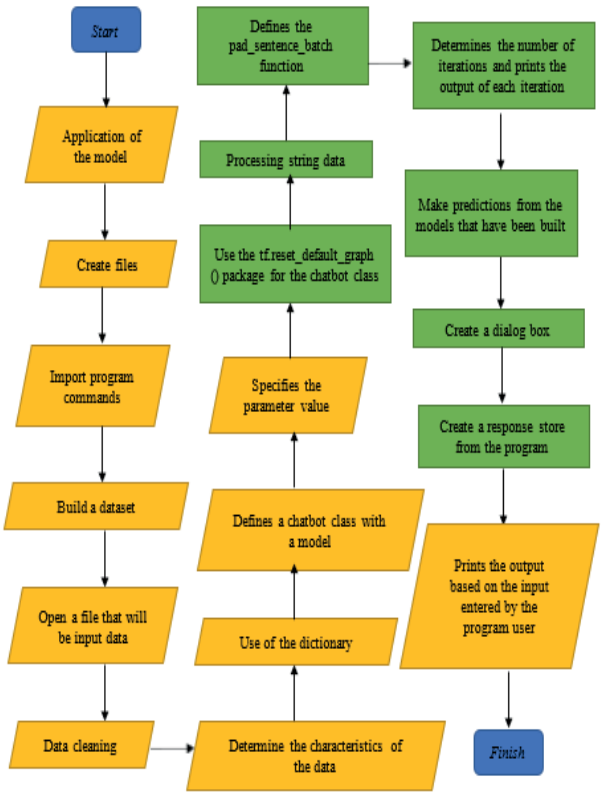

Fig. 5. Program making flowchart

In section V, will be discussed about making the program and discussion of the results which will display the steps for making the program in detail, and the results of the program trial in depth.

V. PROGRAM MAKING AND DISCUSSION OF TEST RESULTS

After the chatbot program has been created, there will be 3 files generated as a result of implementing the LSTM model into the chatbot program, as shown below:

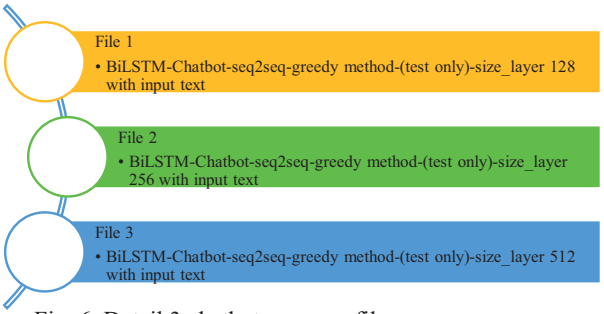

Fig. 6. Detail 3 chatbot program files

TABLE 1. Data to be Tested in BiLSTM Chatbot

Parameter Pair	**File 1**	**File 2**	**File 3**
size_layer	128	256	512
num_layers	2	2	2
embedded_size	64	128	256
learning_rate	0.001, 0.0015	0.001, 0.0015	0.001, 0.0015
batch_size	8	16	32
epoch	20,30, 40,50	20,30, 40,50	20,30, 40,50

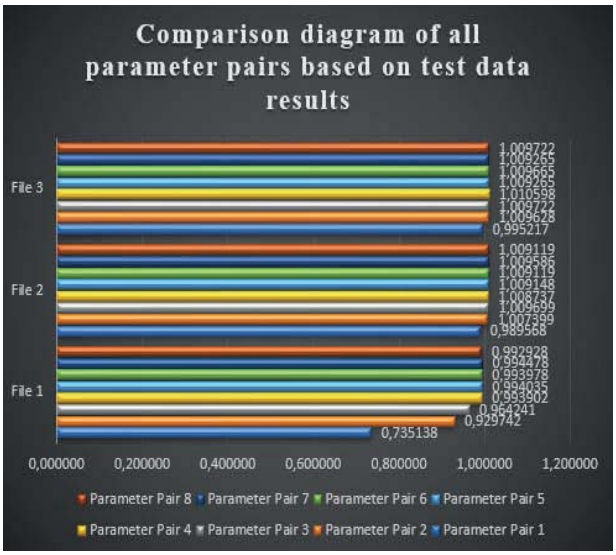

Fig. 7. Comparison diagram of all parameter pairs based on test data results

TABLE 2. Best Test Result Data from Each File

	File 1	**File 2**	**File 3**
Parameter Type	*Parameter Pair 7*	*Parameter Pair 1*	*Parameter Pair 1*
size_layer	128	256	512
num_layers	2	2	2
embedded_size	64	128	256
learning_rate	0.001	0.001	0.001
batch_size	8	16	32
epoch	50	20	20
avg accuracy	0.994478	0.989568	0.995217

Based on the parameter pairs in each of the following files, the best parameter pair from the 3 files that have been tested will not be assessed if > 1.0 an overfit condition occurs (when the training accuracy results are very good but the testing accuracy results are not as good). What will be selected is the one that produces the best average accuracy (avg accuracy) on a scale of 0.0 to 1.0 in the Table 3.

So, based on all the test results of the program that has been carried out, it can be stated that Parameter Pair 1 originating from File 3 is the best parameter pair of the BiLSTM Chatbot with an average accuracy (avg accuracy) value of 0.995217.

The resulting accuracy in the chatbot research based on text dialogue data, the best effect which obtained from original compressed BiLSTM model was 78% accuracy [13]. So when compared to chatbot research with the subject of applying data based on language, this study has increased accuracy

In the section VI, will be delivered a conclusion and future work from this research, based on the results that have been achieved in this research, and things that need to be improved related to this research in the future.

VI. CONCLUSION AND FUTURE WORK

Based on the application of the BiLSTM model into the chatbot, it can be concluded that from all the program test results consisting of a variety of different parameter pairs, it is stated that Parameter Pair 1 (size_layer 512, num_layers 2, embedded_size 256, learning_rate 0.001, batch_size 32, epoch 20) from File 3 is the best parameter pair of the BiLSTM Chatbot using the BiLTSM model, with an average accuracy (avg accuracy) value of 0.995217.

For future works, we can improve the BiLSTM chatbot algorithm using advanced computing environment [14]. In the future work, it is hoped that there will be an increase in the methods, models, and algorithms of the program which will be continuously updated in order to improve the accuracy of the chatbot program based on the compatibility between user input and output produced by the program.

ACKNOWLEDGEMENTS

This research is partially supported by PTUPT 2020 research grant by RISTEKDIKTI with contract number NKB-325/UN2.RST/HKP.05.00/2020.

REFERENCES

[1] S. Raj, Building Chatbots with Python: Using Natural Language Processing and Machine Learning. Apress,: New York City, 2018, p. 33.

[2] A. Bustamam *et al*., "Performance of rotation forest ensemble classifier and feature extractor in predicting protein interactions using amino acid sequences," BMC Genomic, London, 2019, p. 2.

[3] J. Eisenstein, Natural Language Processing, MIT press: Cambridge, 2018, p. 169, p. 345, p.442.

[4] P. Smutny and P. Schreiberova, "Chatbots for learning: A review of educational chatbots for the Facebook Messenger," Elsevier, Amsterdam, 2020, p. 1.

[5] R. Chidananda, "Cornell Movie-Dialogs Corpus," Kaggle datasets, 2018.

[6] D. Shao *et al*., "Domain-Specific Chinese Word Segmentation Based on Bi-Directional Long-Short Term Memory Model," IEEE Access, New Jersey, 2019, p. 12996.

[7] I. Attri and M. Dutta, "Bi-Lingual (English, Punjabi) Sarcastic Sentiment Analysis by using Classification Methods," IEEE Access, New Jersey, 2019, p. 1385.

[8] G. Ardaneswari *et al*., "Implementation of parallel k-means algorithm for two-phase method biclustering in Carcinoma tumor gene expression data," AIP Conference Proceedings, Maryland, 2017, p. 2.

[9] D. Jurafsky and J.H. Martin, Speech and Language Processing An Introduction to Natural Language Processing, Prentice Hall: New Jersey, 2019, p. 66, p. 81, p. 163, p. 497).

[10] W. W. Osterhage, Mathematical Theory of Advanced Computing, Springer Nature: New York City, 2019, p. 51.

[11] J. Loy, Neural Network Projects with Python: The Ultimate Guide to Using Python to Explore the True Power of Neural Networks Through Six Projects, Packt Publishing Ltd: Birmingham, 2019, p. 19.

[12] B. Ru *et al*., "Revisiting the Train Loss: an Efficient Performance Estimator for Neural Architecture Search," Arxiv, New York City, 2020, p. 4.

[13] J. Yin, "A Compression-based BiLSTM for Treating Teenagers' Depression Chatbot," AMMSO, Guilin, 2019, p. 228.

[14] H. Muradi *et al*., "Application of hierarchical clustering ordered partitioning and collapsing hybrid in Ebola Virus phylogenetic analysis," ICACSIS, Depok, 2015, p. 323.