SHM PUBLISHER
SURYA HIJAU MANFAAT

Journal of Soft Computing Exploration

Homepage: https://shmpublisher.com/index.php/joscex

JOSCEX

p-ISSN: 2746-7686
e-ISSN: 2746-0991

.

Real-time detection of Indonesian sign language (ISL) gestures based on long short-term memory

Christy Atika Sari[1], Eko Hari Rachmawanto[2*], Zidan Saifullah[3], Cahaya Jatmoko[4], Daurat Sinaga[5]
[1,2,3,4,5]Study Program in Informatics Engineering, Universitas Dian Nuswantoro, Indonesia

Article Info

Article history:

Received Aug 16, 2024
Revised Sep 5, 2024
Accepted Sep 11, 2024

Keywords:

Long-short-term memory
Indonesian sign language
Multilabel confusion matrix
Mediapipe holistic

ABSTRACT

Deaf people often encounter communication challenges, and sign language serves as a crucial tool for those who cannot speak. In Indonesia, Indonesian Sign Language (ISL) or Sistem Isyarat Bahasa Indonesia (SIBI) is officially recognized by the government and is taught in Special Schools (Sekolah Luar Biasa - SLB). The sign language dictionary comprises 3483 words, facilitating communication and participation in daily life for the deaf community. This research aims to convert ISL gestures within SIBI into understandable text, employing the Long-Short-Term Memory (LSTM) method as the primary approach. The study conducted experiments with two models: Model 1, using a smaller dataset, and Model 2, employing a larger dataset and implementing the k-fold method. The results indicate that Model 2 with k-fold accuracy achieved an accuracy of 98%, while Model 1 reached an accuracy of 85%. Nevertheless, challenges persist in these models, particularly in detecting words with similar gestures, such as'maaf' (sorry) and 'cinta' (love), which may still be misidentified. Despite these challenges, this research contributes positively to the development of assistive technology for the deaf community, enabling more effective communication through sign language.

Corresponding Author:

Eko Hari Rachmawanto,
Study Program in Informatics Engineering, Faculty of Computer Science
Universitas Dian Nuswantoro
Imam Bonjol 207, Semarang, 50131, Central Java, Indonesia
Email: eko.hari@dsn.dinus.ac.id
https://doi.org/10.52465/joscex.v5i3.452

1. INTRODUCTION

Communication is an important process in everyday life, allowing people to convey messages and ideas to others. However, for deaf friends, communicating can be a challenge[1], [2], [3]. These difficulties arise because they may not be able to speak or use spoken language normally. Therefore, the sign language is a very important means of communication for them [4], [5]. In Indonesia, there are two sign language standards used by deaf friends, namely the Indonesian Sign Language System (SIBI) and the Indonesian Sign Language (BISINDO) [6]. By using sign language, they can communicate their thoughts, feelings, and messages clearly and effectively to others, helping them engage in social interactions and gain access to a variety of information and knowledge [7], [8], [9].

SIBI is a sign language standard that has been established by the government of the Republic of Indonesia. SIBI is used in the teaching and learning curriculum in Special Schools (SLB) as a communication tool for deaf students [10]. SIBI was developed by paying attention to the structure and vocabulary of

Indonesian, enabling deaf students to communicate using the sign language related to Indonesian. Currently, in the SIBI dictionary which has been compiled by the Ministry of Education, Culture, Research, and Technology (Kemdikbud), there are 3483 gestures covering various categories, such as alphabetic gestures, affixes, numbers, and words. This dictionary is the result of efforts to expand and enrich the vocabulary of the Indonesian Sign Language System (SIBI), which is the standard sign language used in the teaching and learning curriculum in Special Schools (SLB) [11]. Meanwhile, BISINDO [12] is a sign language that appears naturally in the daily lives of deaf people through interaction with the surrounding environment. BISINDO is more flexible and develops organically according to the communication needs of the deaf community [6]. This sign language is different from SIBI in terms of structure and vocabulary, because it is not strictly related to Indonesian [13]. These two sign languages play an important role in facilitating communication and social integration for deaf friends. With sign language, they can interact, convey messages, and participate in various activities of daily life. As we know, language plays a very important role in forming identity in social life. Communication using the sign language is intended for deaf people, whereas normal people who have never learned the sign language will certainly experience difficulties [14], [15], [16], [17]. However, in an era like today where technological advances, especially in the field of machine learning, have become increasingly advanced, communication is no longer a problem. Because with sign language that uses gestures, technology can recognize these gestures and convert them into a certain output.

In research [7] about an application for translating Indonesian sign language into Android-based voice using TensorFlow. In this research, the convolutional neural network (CNN) algorithm was used to detect the Indonesian sign language. The conclusion was that the accuracy of the model performance was 54.8%. The researcher also suggested that in future research, use the long-short-term memory (LSTM) algorithm. In research [10] which discusses the symbol detection system in SIBI using the Long Short-Term Memory (LSTM) algorithm in real time to detect 6 gestures in SIBI, obtaining accuracy results of 83%. In research [5], research was carried out on the SIBI symbol detection system using a Convolutional Neural Network (CNN). In this study, the detected was 6 symbols, namely 'me', 'you', 'him', 'love', 'sorry', 'sad' with a total dataset of 600 data, with a data split of 90 for training and 10 for testing. Using the CNN model and trained with 250 epochs, an accuracy result of 90% was obtained. Based on previous research., focus on research This is to detect vocabulary gestures in the frequently used Indonesian Sign Language System (SIBI). These gestures were chosen based on the daily communication needs of SIBI users, with the aim of supporting effective communication for deaf friends with friends without hearing loss.

The advantages of this research can be seen by taking the dataset itself, especially if there is no similar data set that has been published on a platform like Kaggle. This can be considered an innovative or novelty aspect in this research. By taking steps to collect its own dataset, this research not only provides a concrete solution to its focus on vocabulary gestures in SIBI but also contributes to the availability of datasets on this topic. The LSTM method was chosen because of its ability to process and recognize temporal patterns in the data. In the context of gesture detection, LSTM has been proven to be effective in learning the temporal sequence of hand movements and recognizing the patterns associated with each vocabulary gesture. Using the LSTM method to detect vocabulary gestures in SIBI, this research is hoped to contribute to the development of a responsive and accurate gesture detection system.

2. METHOD

Long-Short-Term Memory (LSTM)

Long-short-term memory (LSTM) is a modification that has been made to the recurring neural network (RNN) model/method which can make predictions based on past information stored over a long period of time [18], [19], [20]. LSTM has the advantage of remembering and storing past information and being able to study sequential data [21]. In the LSTM model there are three main components, namely the forget gate, the input gate, and the output gate [22]. Each gate has a special function in managing the flow of information in the data sequences as in Figure 1. Based on Figure 1, the following are the equations used in the LSTM method as shown in (1) to (6).

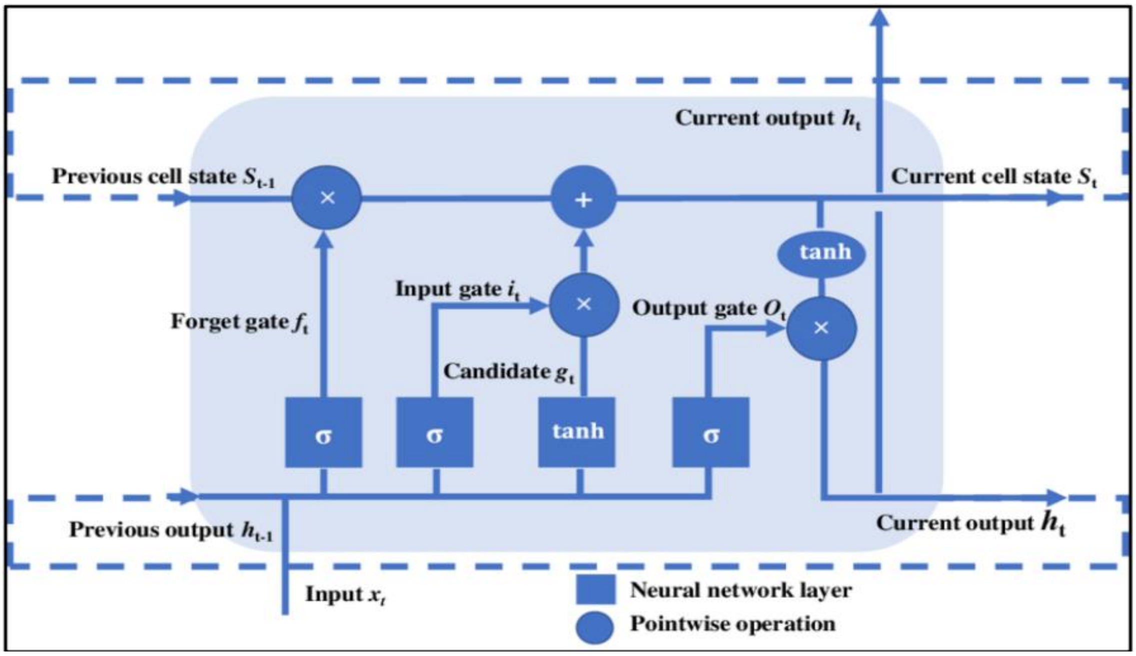


Figure 1. Common LSTM scheme [19], [22]

$$f_t = \sigma\left(w_f \cdot [\,h_{t-1}, x_t] + b_f\,\right) \tag{1}$$
$$i_t = \sigma\left(w_i \cdot [h_{t-1}, x_t] + b_i\,\right) \tag{2}$$
$$S_t^{\sim} = \tanh(w_c \cdot [h_{t-1}, x_t] + b_c\,) \tag{3}$$
$$S_t = f_t * S_{t-1} + i_t * S_t^{\sim} \tag{4}$$
$$o_t = \sigma(w_t \cdot [h_{t-1}, x_t] + b_o\,) \tag{5}$$
$$h_t = o_t * \tanh(S_t) \tag{6}$$

Based on equations (1) to (6), the matrix w is the weight, b is the bias value, and ft , it , St ~, ot , and ht are the results/outputs of the forget gate, input gate, cell state, output gate, and output value at time (t). The forget gate in Long-Short-Term Memory (LSTM) is responsible for deciding whether information in the previous sequence should be ignored or kept stored in LSTM memory. The input gate in LSTM is responsible for determining how much new information should be entered into the LSTM memory. This information is based on the current input and the previous context. The output gate in LSTM Sets how information stored in LSTM memory will be used to produce output at the right time. This gate takes the current input value into account with the previous context to produce the relevant output.

Mediapipe

Mediapipe is a framework that allows developers to create multimodal machine learning pipelines (audio, video), as a landmark. They track the key points of various parts of the body [13]. All keypoint coordinates are normalized as three-dimensional; this is done to provide richer and more accurate information about the position, orientation, and interaction of the body in three-dimensional space, which is useful in various applications such as gesture recognition, pose detection, and body movement analysis. Mediapipe Holistic uses pose, face, and hand landmark models which produce 468 face keypoints,, 21 hand keypoints and 33 pose keypoints as in Figures 2 and Figure 3.

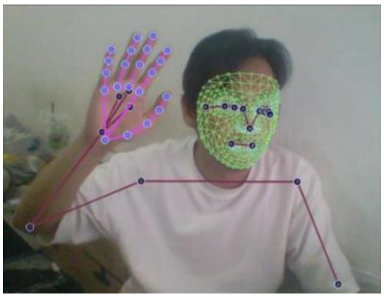


Figure 2. Keypoint sample result

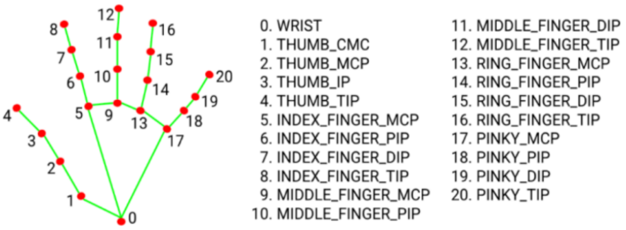


Figure 3. Hand landmark

Multilabel Confusion Matrix

The Multilabel Confusion Matrix is a method that can be used to measure or analyze the performance of a classification/detection model. The multilabel confusion matrix has information that compares the results performed by the system with the actual results. When using a multilabel confusion matrix, there are four terms that represent the results of the evaluation process. The four terms are True Positive (TP), True Negative (TN), False Positive (FP), and False Negative. By using the multi-label confusion matrix, we can calculate various evaluation metrics such as precision, recall, and F1 score for each label separately. This helps to gain a deeper understanding of the model's performance in recognizing each label individually on multi-label classification tasks.

Face Dataset

This data set was created to support research into gesture recognition in sign language using the LSTM model. This data set consists of gesture frames in a series of sign language that have been converted into Numpy array format (.npy) as in Figure 4. Each sign language gesture video will have 30 frames and each vocabulary label will have a total of 80 videos according to Figure 5. The following is a detailed explanation of this dataset:

1. Data Format. Each video frame is represented in the form of a Numpy array, which includes keypoint information extracted using MediaPipe Holistic. The keypoint information includes landmark positions of the hand, face, and body pose.
2. Folder Structure. The dataset is organized in a hierarchical folder structure. This folder structure includes the following:
   - The Root Folder is the main folder that contains all the datasets
   - Folder Labels, each vocabulary label has its own folder, such as 'I', 'You', 'He', 'Love', ‘Sorry’, and 'Sad'.
   - Video Folder, each vocabulary label has a video, each saved in an associated folder.
   - Numpy Frames, each video folder contains 30 frames, and each frame is represented in numpy array format (.npy)
3. Total amount of data. The total amount of data in the dataset is calculated by multiplying the number of vocabulary labels, the number of videos per label, and the number of frames per video. In this case, the total data is the result of 6 labels x 80 videos per x label, where 30 frames per video = 14,400 frames.
4. Movement Labels and Variations. Each vocabulary label represents one word in sign language, such as 'I', 'You', 'He', 'Love', 'Sorry', 'Sad'. By providing 80 videos per label, the dataset includes variations in movement for each word, including variations in body pose and facial expressions.
5. Intended Use. This dataset is designed to train and test an LSTM model in recognizing sign language gestures. The training process is carried out using data from most videos, while data that have never been seen before is used for model testing to measure generalization as test data. This dataset is expected to provide a strong basis for the development of responsive and accurate sign language gesture recognition models.

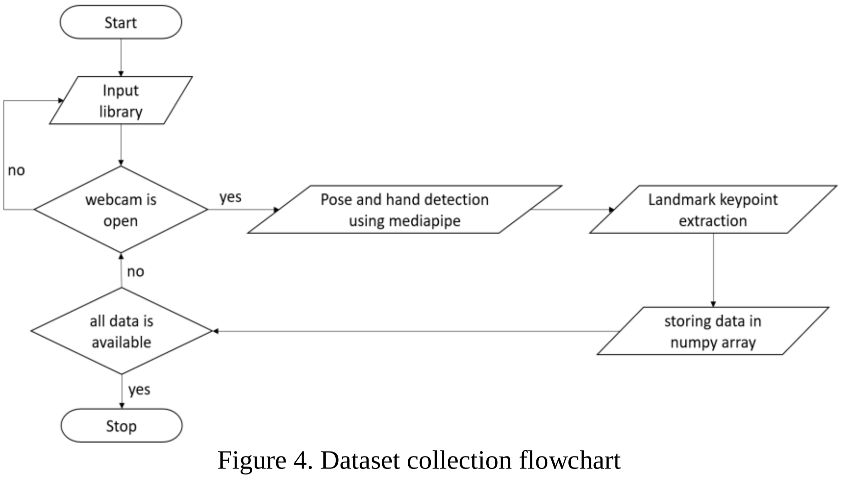

Start
Input library
no
webcam is open
yes
Pose and hand detection using mediapipe
Landmark keypoint extraction
no
all data is available
storing data in numpy array
yes
Stop

Figure 4. Dataset collection flowchart

.

After the dataset has been collected, the next step is to preprocess the data to prepare the dataset before the training and testing process is carried out. One of the stages in data preprocessing is creating labels or action mapping from the video image dataset that has been collected. Each action in the dataset will be given a label based on the type of sentence it represents. The following are the label-mapping results for each action:

a) The 0th index shows the action of the word "I"
b) The first index shows the action of the word "you"
c) The second index shows the action of the word "he"
d) The 3rd index shows the action of the word "love"
e) The 4th index shows the action of the word "sorry"
f) The 5th index shows the action of the word “sad”.

Then, from the six data labels, the data was divided into training data and testing data with a percentage of 80% train and 20% val. The 80:20 data division was chosen because it uses 80% of the data as training data; the model can learn the patterns and relationships in the data well because it has many samples to process. Additionally, to ensure the objectivity of the model evaluation, the researcher also decided to separate the test data separately.

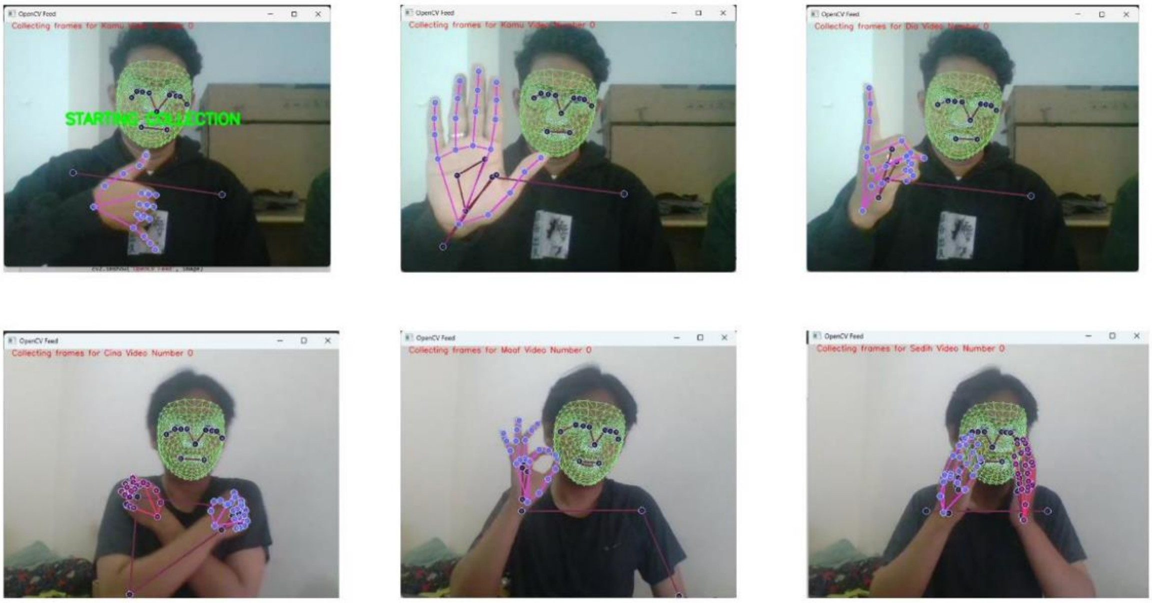

OpenCV Feed
Collecting frames for Kamu Video Number 0
STARTING COLLECTION
OpenCV Feed
Collecting frames for Kamu Video Number 0
OpenCV Feed
Collecting frames for Dia Video Number 0
OpenCV Feed
Collecting frames for Cina Video Number 0
OpenCV Feed
Collecting frames for Maaf Video Number 0
OpenCV Feed
Collecting frames for Sedih Video Number 0

Figure 5. Sample data set

Proposed Method

The workflow of the SIBI gesture detection method consists of several important steps as in Figure 6. These stages are designed to produce a model that can recognize SIBI gestures with high accuracy. First, the workflow starts with the keypoint initialization stage. At this stage, the function is to save important points of the face, pose, and hands that have been taken using MediaPipe Holistic. This function will be the basis for the feature extraction and data collection process. After that, the next step is to extract key points. At this stage, important points that have been taken using MediaPipe Holistic are used to produce numerical features that represent SIBI gestures.

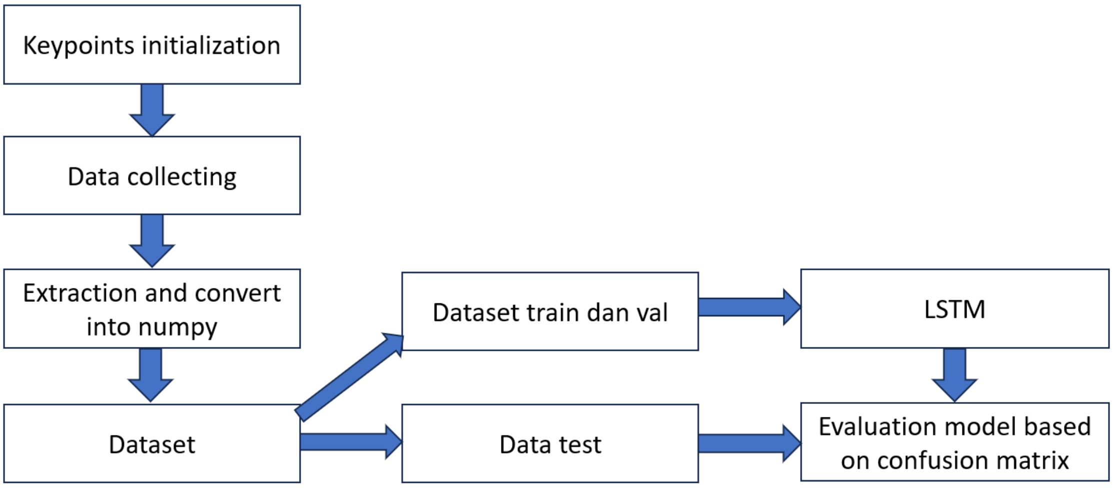


Figure 6. Indonesian sign language (ISL) proposed scheme

The spatial information from the key points is converted into a numerical representation that can be used by machine learning models. After extracting keypoints, the keypoint data collection process is carried out. At this stage, a dataset consisting of SIBI gestures and extracted numerical features is collected. These data will become training material for machine learning models. Next, the data are divided into two parts, training data and test data in the split data stage. Training data will be used to train the model, while test data will be used to test the performance of the model that has been trained. After the data are shared, the next stage is the modeling process. At this stage, machine learning models, such as LSTM (Long Short-Term Memory), are built using training data. This model will learn and study SIBI gesture patterns. Finally, the evaluation stage is performed to measure the performance of the model that has been built. The model will be evaluated using test data to measure the accuracy and performance of SIBI gesture detection.

Initialize Function for Keypoints

Before collecting data, researchers initialized keypoints for the face, pose, left hand and right hand using the OpenCV and MediaPipe Holistic libraries. The OpenCV library is used to record movement directly via webcam, while MediaPipe Holistic is used to detect and track keypoints on the face, body pose, left-hand and right hand. The initialization process begins by importing the OpenCV and MediaPipe Holistic libraries into the project. The researchers used the functions provided by OpenCV to access the webcam and capture video frames in real-time. Each video frame is then processed using MediaPipe Holistic to detect and track the necessary keypoints. For facial keypoints, MediaPipe Holistic will identify important points such as eyes, nose, lips, and others. Meanwhile, for body pose keypoints, MediaPipe Holistic will recognize body position and orientation, including points such as shoulders, elbows, wrists, and others. The left-hand and right-hand keypoints will also be identified by MediaPipe Holistic by detecting hand position and movement. After the keypoints were successfully identified and tracked, the researcher used previously defined functions to store the facial, body pose, left hand, and right hand keypoint data in the appropriate data structure.

These keypoint data will be used in the next stage for analysis and movement recognition in gesture detection using the Indonesian Language Sign System (SIBI). By initializing keypoints using OpenCV and MediaPipe Holistic, researchers can obtain very important information regarding the position and movement of faces, body poses, and hands in gesture detection. This initialization is a crucial first step in the data collection and gesture analysis process using the Indonesian Sign Language System (SIBI).

LSTM Modeling

In the modeling stage in real-time SIBI detection, the next step after data division is to build a long-short-term memory (LSTM) model. The LSTM (Long Short-Term Memory) model is used to learn patterns and relationships between SIBI gesture frames with the aim of accurately detecting gestures. LSTM is a type of recurrent neural network that has the ability to remember long-term information and handle long-term dependency problems. The flow in the modeling process begins by determining the architecture of the LSTM model consisting of several layers, including the first LSTM layer with 256 units and return_sequences=True, followed by a dropout layer. The second LSTM layer with 128 units, and return_sequences=True, is added next, followed by another dropout layer to avoid overfitting. In the next stage, there is a third LSTM layer with 64 units and again followed by a dropout layer. Dropout layers are applied after each LSTM layer to prevent

.

random overfitting. This entire structure is designed with the aim of obtaining a model that can capture patterns in the data and produce accurate predictions in the context of the classification task at hand. Finally, a dense layer with softmax activation is used as the output layer, which produces gesture predictions based on the given input. By building this LSTM model, the algorithm will undergo a training phase using the previously divided data. During the training process, the model will iteratively adjust its internal parameters, including the weight W and bias, based on the patterns and relationships present in the training data. This process is done by minimizing the difference between the predictions generated by the model and the actual values 27 in the training data. The model learns to recognize temporal patterns and relationships between variables in the data, allowing it to produce accurate predictions. Once the training process is completed, the adjusted LSTM model will be ready to use. When performing gesture prediction in real-time, new inputs are presented to the model, and the model will use its knowledge gained during training to produce predictions according to the observed gestures.

3. RESULTS AND DISCUSSIONS

In this chapter, the results of previous research will be explained, namely the discussion of the results of model training using the Long Short-Term Memoryu LSTM algorithm, and continued with the calculation of values in the evaluation using the confusion matrix, namely the recall value, precision, and f1 score. In this study, the researchers conducted trials using 2 models with the same algorithm. However, there is a difference in the number of datasets where model 1 has fewer datasets than model 2, then there is also a difference in the data split stage, where model 1 will use the data split method using the function from the sklearn library, namely train_test_split(), while model 2 uses cross-validation KFold. The following is a table of the differences between Model 1 and model 2.

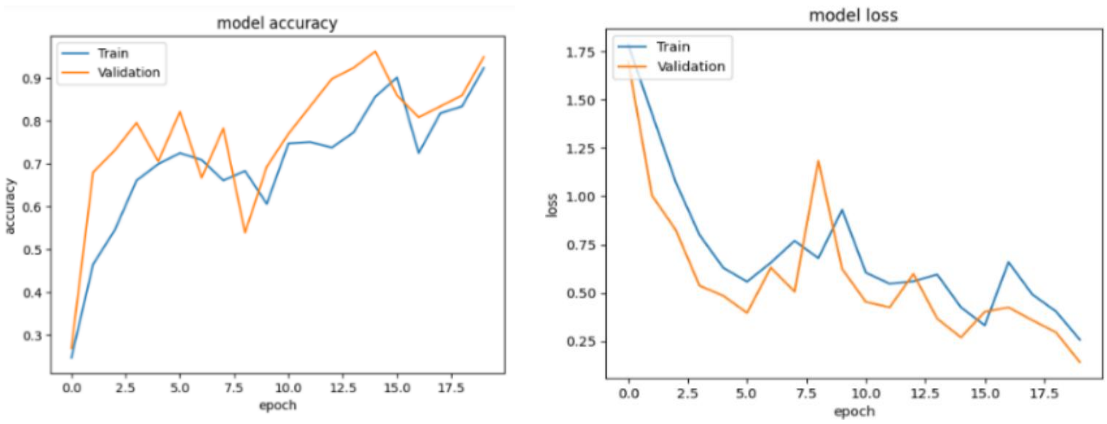
Before entering the data processing stage, the first process is carried out to take and prepare the data set which is the basis for developing the model. The data set used consists of two models, each of which includes a number of videos representing hand gestures in the Indonesian Sign Language System (SIBI). The first model contains 390 videos, while the second model has 480 videos. First, model 1 was tested using the code below.

```
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2)
from tensorflow.keras.layers import LSTM, Dense, Dropout
model = Sequential() model.add(LSTM(units=256,
return_sequences=True,
input_shape=(timesteps, input_dim),
bias_initializer=bias_init)) model.add(Dropout(0.2))
model.add(LSTM(units=128, return_sequences=True;
bias_initializer=bias_init)) model.add(Dropout(0.2))
model.add(LSTM(units=64,
bias_initializer=bias_init))
model.add(Dense(actions.shape[0], activation='softmax'))
```

The testing process began by dividing the model 1 data set into two subsets, training data and validation data. The data division ratio is 80% for training and 20% for validation. This is done to train the model using fairly representative data and then test the model's performance on data that have never been seen before. After splitting the data, the next step is to create a model using the LSTM algorithm. Model formation is done by importing the LSTM, Dense, and Dropout layers from the tf.keras library. The addition of dense and Dropout layers serves to reduce overfitting. This model will be trained with SIBI gesture data. The training process is carried out in several stages called epochs; epochs are a one-time training process on all data. The model will be trained 20 times, which means the model will see and update its knowledge from the training data 20 times as in Table 1. The following is a visualization of the training performance of model 1 with the LSTM algorithm without using KFold as in Figure 7. After going through the model training process, the next step is to evaluate the model's performance by utilizing the confusion matrix. The prediction results obtained from the model will be carefully compared with the actual labels on the test data as in Table 2. To measure the accuracy and effectiveness of the model, relevant evaluation metrics are used, such as accuracy, precision, recall, and F1-score. By conducting an in-depth analysis of the confusion matrix and these evaluation metrics, we can understand the extent to which the model is capable of carrying out the classification task correctly and accurately, providing a better picture of quality of the performance of the model built.

Table 1. Accuracy result based on epoch using the 1st model

Epoch	**Accuracy**	**Loss**	**Val accuracy**	**Val loss**
1	0.2269	1.8213	0.2692	1.6982
5	0.7234	0.5832	0.7051	0.4850
10	0.5870	1.0436	0.6923	0.6251
15	0.8607	0.4542	0.9615	0.2692
20	0.9053	0.2789	0.9487	0.1412



(a) Model Accuracy (b) Model loss

Figure 7. Training and validation graph for accuracy and loss

Table 2. Performances based on the 1st model

Facial expression	Precision	Recall	F1-score	Support
I	1.00	0.67	0.80	15
You	1.00	0.47	0.64	15
He/she	1.00	0.87	0.93	15
Love	0.75	1.00	0.86	15
Sorry	0.68	1.00	0.81	15
Sad	1.00	1.00	1.00	15
Macro Avg	0.91	0.83	0.84	90
Weighted Avg	0.91	0.83	0.84	90

The second model was tested using a dataset that had a larger number compared to the dataset for model 1, and using KFold. First, a model was created using the LSTM algorithm which is similar to the 1st model. Model 2 has been tested using the following code.

```
model = Sequential()
model.add(LSTM(units=256,
return_sequences=True,
input_shape=(timesteps,
input_dim),
bias_initializer=bias_init))
model.add(Dropout(0.2))
model.add(LSTM(units=128,
return_sequences=True;
bias_initializer=bias_init))
model.add(Dropout(0.2))
model.add(LSTM(units=64,
bias_initializer=bias_init))
model.add(Dense(actions.shape[0], activation='softmax'))
k = 5 # jumlah fold kfold = KFold(n_splits=k, shuffle=True)
```

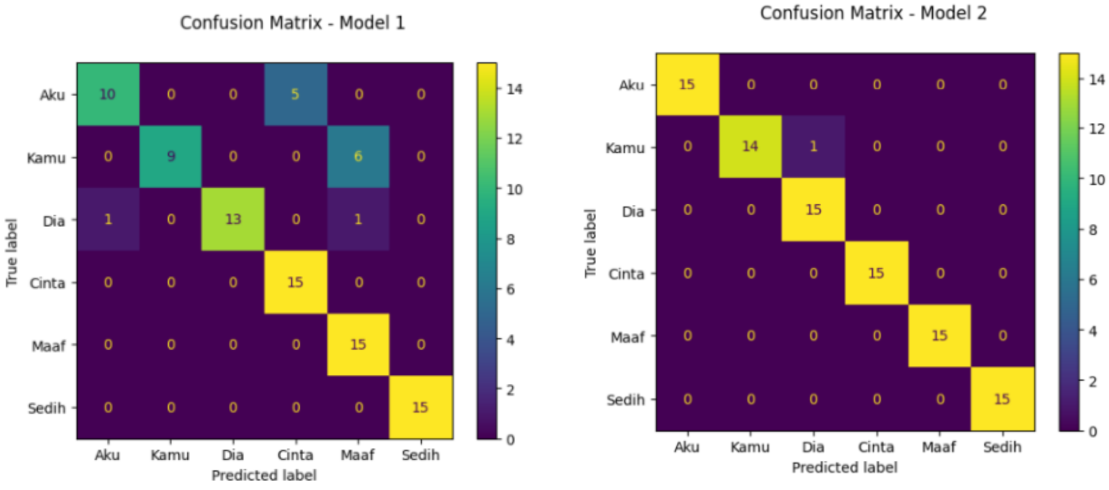
After creating the LSTM algorithm model, KFold was set up using 5 folds. KFold is a cross-validation method that divides a data set into k subsets (in this case 5 subsets). Each subset will be used as test data in turn, while the other subset will be used as training data, so that each data has the same opportunity as training data and val data. Then the training process was carried out for the second model, using epochs 30 times and KFold with folds 5 times. Therefore, each fold will carry out a 30 epoch training process, as in Table 3.

Table 3. Performances based on the second model

Facial expression	Precision	Recall	F1-score	Support
I	1.00	1.00	1.00	15
You	1.00	0.93	0.97	15
He/she	0.94	1.00	0.97	15
Love	1.00	1.00	1.00	15
Sorry	1.00	1.00	1.00	15
Sad	1.00	1.00	1.00	15
Macro Avg	0.99	0.99	0.99	90
Weighted Avg	0.99	0.99	0.99	90

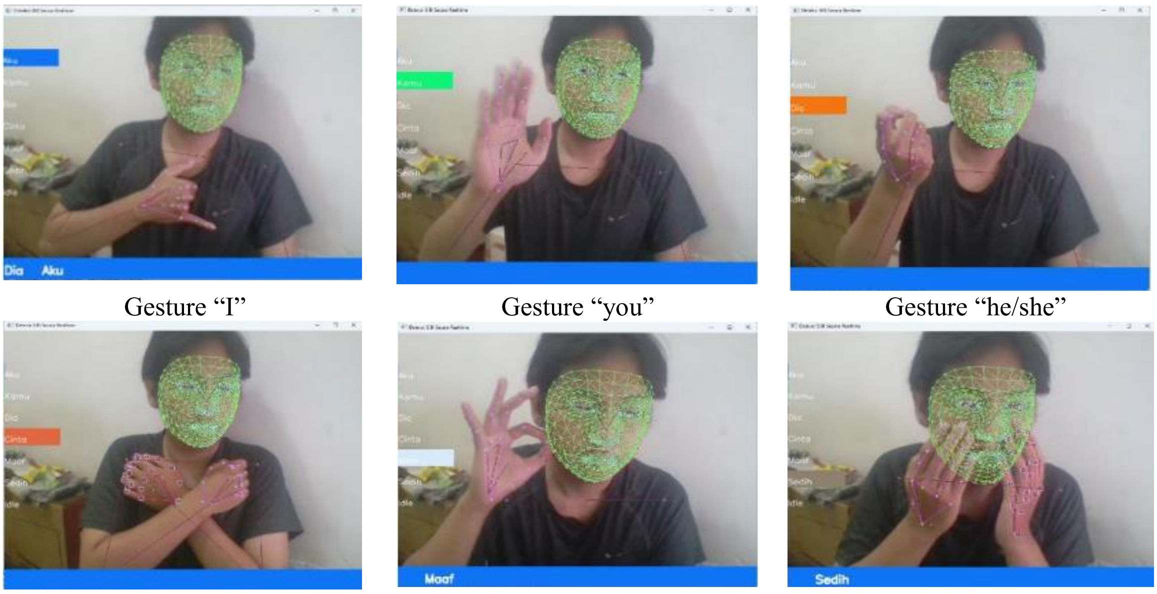
The test results in Table 3 not only provide information on the reliability of the LSTM model in hand gesture detection in general, but also reveal the potential of the model in dealing with more complex usage scenarios and larger amounts of data. By involving a data set that includes a variety of SIBI hand gestures and testing it outside the training sample, this evaluation provides a deep understanding of the extent to which the model can adapt to situations it has never encountered before. The information obtained from this testing will help identify the model's strengths and weaknesses, providing a more holistic view of the potential use of the LSTM model in Indonesian sign language hand gesture detection applications. Additionally, involving more complex usage scenarios can also provide better insight into the limitations of the model, opening up opportunities for further improvement and development in the future. In this research, the model was evaluated using testing data that were never included in the model training process. To quantitatively measure model performance, a comparison was made between the first model and the second model by looking at accuracy and loss using the model.evaluate() method. The comparison table for the accuracy and loss of the model evaluation was seen in Table 4.

Furthermore, the researchers also used the multilabel confusion matrix (MCM) evaluation method to analyze the results of SIBI gesture detection in more detail. The use of MCM was chosen because the dataset used has different labels, so this method is more suitable for describing prediction accuracy in multilabel cases. The table used in MCM is slightly different from the usual confusion matrix table used in the case of binary classification. In the scikit-learn library documentation, there is a complete explanation of the multi-label confusion matrix and the table format used. The MCM table consists of rows representing actual labels and columns representing predicted labels. Each cell in the table stores the number of samples that fall into a particular category based on the actual label and the predicted label. By using the MCM, researchers can calculate evaluation metrics such as precision, recall, F1 score and accuracy which are useful for measuring the model performance in accurately predicting SIBI gestures as in Figure 8. By using the MCM evaluation method, researchers can describe more comprehensively the model's ability to recognize and differentiate between different SIBI gestures. In-depth analysis of the MCM tables will provide a better understanding of the model performance and allow researchers to identify areas that need improvement in SIBI's real-time gesture detection process. The gesture samples are shown in Figure 9.



Confusion matrix 1st model Confusion matrix 2nd model

Figure 8. A comparison of confusion matrix



Gesture "I" Gesture "you" Gesture "he/she"

Gesture "love" Gesture "sorry" Gesture "sad"

Figure 9. Samples gestures detected for real-rime detection

On the other hand, the results of model detection that have gone through the training and testing process. Researchers conducted trials using the second model, researchers carried out real-time detection trials 9 times with 3 different subjects. revealed the results of a series of real-time detection trials that had been carried out by researchers 9 times. Success and failure in detecting sign vocabulary gestures are apparently influenced by various factors that need to be considered. One of the crucial factors is the similarity of one gesture vocabulary to another, where detected gestures must be differentiated correctly. Additionally, factors such as light also play a significant role. For example, a'sorry' gesture can be detected as a 'you' gesture when lighting conditions are less than optimal. Despite facing these challenges, the successful detection accuracy obtained after conducting nine trials showed 92% results. These results reflect the level of accuracy of the system in recognizing and classifying gestures from the sign vocabulary in real-time situations. The following are the results of the real-time vocabulary detection.

4. CONCLUSION

This study shows that the LSTM model with a larger dataset (Model 2) and the application of k-fold cross validation with 30 epochs for each fold can produce the best accuracy in detecting Indonesian Sign Language System (SIBI) gestures, reaching an average value of 98%. On the contrary, the LSTM model in Model 1, which uses a smaller number of data sets and without k-fold validation, is only able to achieve an accuracy of 85% with 20 epochs of training. Thus, it can be concluded that the use of k-fold cross-validation and the use of a larger dataset can improve the accuracy of gesture detection in the LSTM model. Successfully applying the LSTM algorithm for real-time SIBI gesture detection. The following are the results of the direct test with each gesture tested 9 times on 3 different subjects. The gesture 'I' gets 100% accuracy. The gesture 'you' gets 100% accuracy. The gesture 'he' gets 100% accuracy. The gesture 'love' gets 88.89% accuracy. The gesture'sorry' gets 66.67% accuracy. The gesture'sad' gets 100% accuracy. In further research, it can optimize accuracy gain by adding datasets for vocabulary from the Indonesian Sign Language System (SIBI), which are more extensive, adding variations of objects and different places during data collection.

REFERENCES

[1] D. Pribadi, M. Wahyudi, D. Puspitasari, A. Wibowo, R. Saputra, and R. Saefurrohman, "Real Time Indonesian Sign Language Hand Gesture Phonology Translation Using Deep Learning Model," in *Proceedings of the 3rd International Conference on Advanced Information Scientific Development*, SCITEPRESS - Science and Technology Publications, Mar. 2023, pp. 172–176. doi: 10.5220/0012446000003848.

[2] A. Josef and G. P. Kusuma, "Alphabet Recognition in Sign Language Using Deep Learning Algorithm with Bayesian Optimization," *Revue d'Intelligence Artificielle*, vol. 38, no. 3, pp. 929–938, Jun. 2024, doi: 10.18280/ria.380319.

[3] E. Rakun, A. M. Arymurthy, L. Y. Stefanus, A. F. Wicaksono, and I. W. W. Wisesa, "Recognition of Sign Language System for Indonesian Language Using Long Short-Term Memory Neural Networks," *Adv Sci Lett*, vol. 24, no. 2, pp. 999–1004, Feb. 2018, doi: 10.1166/asl.2018.10675.

[4] A. A. Pratama, E. Rakun, and D. Hardianto, "Human Skeleton Feature Extraction from 2-Dimensional Video of Indonesian Language Sign System (SIBI [Sistem Isyarat Bahasa Indonesia]) Gestures," in *Proceedings of the 2019 5th International Conference on Computing and Artificial Intelligence*, New York, NY, USA: ACM, Apr. 2019, pp. 100–105. doi: 10.1145/3330482.3330484.

[5] N. F. P. Setyono and E. Rakun, "Recognizing Word Gesture in Sign System for Indonesian Language (SIBI) Sentences Using DeepCNN and BiLSTM," in *2019 International Conference on Advanced Computer Science and information Systems (ICACSIS)*, IEEE, Oct. 2019, pp. 199–204. doi: 10.1109/ICACSIS47736.2019.8979772.

[6] D. Sujatmiko, C. A. Sari, E. H. Rachmawanto, A. D. Krismawan, B. R. Altamer, and M. A. Alkhafaji, "AlexNet Architecture Based Convolution Neural Network for Realtime Audio to Text Translator of Bisindo Hand Sign," in *2023 International Seminar on Application for Technology of Information and Communication (iSemantic)*, IEEE, Sep. 2023, pp. 429–434. doi: 10.1109/iSemantic59612.2023.10295322.

[7] N. Hikmatia and M. I. Zul, "Aplikasi Penerjemah Bahasa Isyarat Indonesia menjadi Suara berbasis Android menggunakan Tensorflow," *Jurnal Komputer Terapan*, vol. 7, no. 1, pp. 74–83, 2021, [Online]. Available: https://jurnal.pcr.ac.id/index.php/jkt/

[8] S. B. Abdullahi and K. Chamnongthai, "American Sign Language Words Recognition of Skeletal Videos Using Processed Video Driven Multi-Stacked Deep LSTM," *Sensors*, vol. 22, no. 4, p. 1406, Feb. 2022, doi: 10.3390/s22041406.

[9] E. Maryadi, S. Syahrul, D. Maulidya, R. Risnandar, E. Prakasa, and D. Andriana, "Hand Skeleton Graph Feature for Indonesian Sign Language (BISINDO) Recognition Based on Computer Vision," in *Proceedings of the 2022 International Conference on Computer, Control, Informatics and Its Applications*, New York, NY, USA: ACM, Nov. 2022, pp. 256–260. doi: 10.1145/3575882.3575931.

[10] J. Sulaksono, I. A. Dwi Girinatari, M. Sudarma, and I. B. Alit Swarmardika, "SIBI Syllable Recognition System With LSTM," in *2023 International Conference on Smart-Green Technology in Electrical and Information Systems (ICSGTEIS)*, IEEE, Nov. 2023, pp. 94–97. doi: 10.1109/ICSGTEIS60500.2023.10424084.

[11] I. D. M. B. A. Darmawan *et al.*, "Advancing Total Communication in SIBI: A Proposed Conceptual Framework for Sign Language Translation," in *2023 International Conference on Smart-Green Technology in Electrical and Information Systems (ICSGTEIS)*, IEEE, Nov. 2023, pp. 23–28. doi: 10.1109/ICSGTEIS60500.2023.10424020.

[12] N. Ahmad, E. S. Wijaya, C. Tjoaquinn, H. Lucky, and I. A. Iswanto, "Transforming Sign Language using CNN Approach based on BISINDO Dataset," in *2023 International Conference on Informatics, Multimedia, Cyber and Informations System (ICIMCIS)*, IEEE, Nov. 2023, pp. 543–548. doi: 10.1109/ICIMCIS60089.2023.10349011.

[13] Y. D. Maheswara, M. A. Al-Sulthon, P. A. Wicaksono, K. Afifah, and N. Prihatiningrum, "Real-Time BISINDO Sign Language Recognition: A Dynamic Approach with GRU and LSTM Models Leveraging MediaPipe," in *2023 6th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, IEEE, Dec. 2023, pp. 226–232. doi: 10.1109/ISRITI60336.2023.10467586.

[14] M. Y. Daffa Izzalhaqqi and Wahyono, "Gesture Recognition in Indonesian Sign Language Using Hybrid Deep Learning Models," in *2023 International Workshop on Intelligent Systems (IWIS)*, IEEE, Aug. 2023, pp. 1–6. doi: 10.1109/IWIS58789.2023.10284666.

[15] N. F. P. Setyono and E. Rakun, "Recognizing Word Gesture in Sign System for Indonesian Language (SIBI) Sentences Using DeepCNN and BiLSTM," in *2019 International Conference on Advanced Computer Science and information Systems (ICACSIS)*, IEEE, Oct. 2019, pp. 199–204. doi: 10.1109/ICACSIS47736.2019.8979772.

[16] G. Kusuma Atmaja, H. Hikmayanti, and S. Faisal, "Object Detection of Indonesian Sign Language System Using Yolov7 Method Deteksi Objek Sistem Isyarat Bahasa Indonesia Dengan Metode YOLOV7," *Jurnal Teknik Informatika (JUTIF)*, vol. 5, no. 4, pp. 1197–1203, 2024, doi: 10.52436/1.jutif.2024.5.4.2468.

[17] D. Pribadi, M. Wahyudi, D. Puspitasari, A. Wibowo, R. Saputra, and R. Saefurrohman, "Real Time Indonesian Sign Language Hand Gesture Phonology Translation Using Deep Learning Model," in *Proceedings of the 3rd International Conference on Advanced Information Scientific Development*,

SCITEPRESS - Science and Technology Publications, Mar. 2023, pp. 172–176. doi: 10.5220/0012446000003848.

[18] N. Amangeldy, I. Krak, B. Kurmetbek, and N. Gazizova, “A Comparison of the Effectiveness Architectures LSTM1024 and 2DCNN for Continuous Sign Language Recognition Process,” in *Seventh International Workshop on Computer Modeling and Intelligent Systems*, 2024.

[19] - Ridwang, A. A. Ilham, I. Nurtanio, and - Syafaruddin, “Dynamic Sign Language Recognition Using Mediapipe Library and Modified LSTM Method,” *Int J Adv Sci Eng Inf Technol*, vol. 13, no. 6, pp. 2171–2180, Dec. 2023, doi: 10.18517/ijaseit.v13i6.19401.

[20] R. E. Caraka *et al.*, “Empowering deaf communication: a novel LSTM model for recognizing Indonesian sign language,” *Univers Access Inf Soc*, Mar. 2024, doi: 10.1007/s10209-024-01095-1.

[21] B. A. Wisesa, W. Andriyani, and B. D. P. Purnomosidi, “Usage of LSTM Method On Hand Gesture Recognition For Easy Learning of Sign Language Based On Desktop Via Webcam,” in *2022 5th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, IEEE, Dec. 2022, pp. 148–153. doi: 10.1109/ISRITI56927.2022.10053076.

[22] A. A. Ilham, I. Nurtanio, Ridwang, and Syafaruddin, “Applying LSTM and GRU Methods to Recognize and Interpret Hand Gestures, Poses, and Face-Based Sign Language in Real Time,” *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 28, no. 2, pp. 265–272, Mar. 2024, doi: 10.20965/jaciii.2024.p0265.

.