



UNIVERSITAS
MERCU BUANA

**BISINDO INDONESIAN SIGN LANGUAGE RECOGNITION USING
MEDIAPIPE HOLISTIC AND LSTM DEEP LEARNING MODEL**

Thesis Report

UNIVERSITAS
MERCU BUANA

Robby Auziqni
41518010085

**DEPARTMENT OF INFORMATICS
FACULTY OF COMPUTER SCIENCE
UNIVERSITAS MERCU BUANA
JAKARTA
2022**



UNIVERSITAS
MERCU BUANA

**BISINDO INDONESIAN SIGN LANGUAGE RECOGNITION USING
MEDIAPIPE HOLISTIC AND LSTM DEEP LEARNING MODEL**

Thesis Report

Submitted to complete one of the requirements
to get a Bachelor's Degree in Computer Science

UNIVERSITAS
MERCU BUANA

Robby Auziqni
41518010085

**INFORMATICS DEPARTMENT
FACULTY OF COMPUTER SCIENCE
UNIVERSITAS MERCU BUANA
JAKARTA
2022**

ORIGINALITY STATEMENT SHEET

The undersigned below:

Student Id : 41518010085
Name : Robby Auziqni
Thesis Title : BISINDO Indonesian Sign Language Recognition Using
MediaPipe Holistic and LSTM Deep Learning Model

Stating that my final project report is **my work and not plagiarism**. If it is found in my final project report that there is an element of plagiarism, then I am ready to get academic sanctions related to this.



Jakarta, 15 February 2022



Robby Auziqni

UNIVERSITAS
MERCU BUANA

THESIS COPYRIGHT LICENSE

As a student of Universitas Mercu Buana, I the undersigned below:

Name : Robby Auziqni
Student Id : 41518010085
Thesis Title : BISINDO Indonesian Sign Language Recognition
Using MediaPipe Holistic and LSTM Deep
Learning Model

Hereby give permission and agreed to give Universitas Mercu Buana Non-exclusive Royalty-Free Right for my scientific work entitled above along with existing tools (if necessary).

With this **Non-exclusive Royalty-Free Right**, Universitas Mercu Buana has the right to store, transfer, re-format, manage in the form of a database, maintain and publish my thesis.

In addition, for the scientific development within the Universitas Mercu Buana, I permit the Researcher at the Research Lab of the Faculty of Computer Science, Universitas Mercu Buana to use and develop the research results in this thesis for research and publication purposes as long keep my name as the author/creator and as the owner of the Copyright.

Thus, I made this statement in truthfulness, and it is used as a requirement in Submitting this Thesis.

Jakarta, 15 February 2022




Robby Auziqni

THESIS PUBLICATION STATEMENT

As a student of Universitas Mercu Buana, I the undersigned below:

Name : Robby Auziqni
 Student Id : 41518010085
 Thesis Title : BISINDO Indonesian Sign Language Recognition
 Using MediaPipe Holistic and LSTM Deep
 Learning Model

State that:

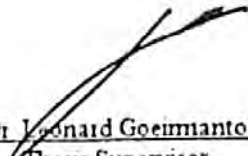
1. The publication of my thesis is as follows:

No	Publication	Type		Status	
		1	Scientific Publication	Jurnal Nasional Tidak Terakreditasi	
Jurnal Nasional Terakreditasi	✓				
Jurnal International Tidak Bereputasi					
Jurnal International Bereputasi					
1	Submitted/published in:	Journal Name			
		ISSN			
		Journal URL			
		File URL (if published)			

2. Willing to complete the entire process of publication, starting from submitting, revising until it is declared can be published in the intended journal.
3. Requested to attach a scan of your KTP and Statement Letter (See Attachment to the Intellectual Property Rights Document), to register if needed.

Thus, I made this statement true.

Acknowledgement,
 Thesis Supervisor


 (Dr. Leonard Goeimanto)
 Thesis Supervisor

Jakarta, 15 February 2022



Robby Auziqni

APPROVAL SHEET

Name : Robby Auziqni
Student Id : 41518010085
Thesis Title : BISINDO Indonesian Sign Language Recognition
Using MediaPipe Holistic and LSTM Deep
Learning Model

This Thesis has been reviewed and approved for Thesis Defence.

Jakarta, 7 July 2022



EXAMINER APPROVAL SHEET

Name : Robby Auziqni
Student Id : 41518010085
Thesis Title : BISINDO Indonesian Sign Language Recognition
Using MediaPipe Holistic and LSTM Deep
Learning Model

This Final Project has been examined in **Thesis Defence** as one of the requirements to obtain a Bachelor's degree in the Informatics Department, Faculty of Computer Science, Universitas Mercu Buana.

Jakarta, 14 July 2022

Approved By,



UN (Dr. Ida Nurhaida, M.T)
Chief Examiner

MERCU BUANA



(Vina Ayumi, M.Kom)
Examiner 2



(Prastika Indriyanti, S.Kom, M.Cs)
Examiner 3


COMMITTEE APPROVAL SHEET

Name : Robby Auziqni
Student Id : 41518010085
Thesis Title : BISINDO Indonesian Sign Language Recognition Using
MediaPipe Holistic and LSTM Deep Learning Model

This Final Project has been examined in **Thesis Defence** as one of the requirements to obtain a Bachelor's degree in the Informatics Department, Faculty of Computer Science, Universitas Mercu Buana.

Jakarta, 1st August 2022

Approved by,


(Dr. Leonard Goeimanto)
Thesis Supervisor

Acknowledged by,

UNIVERSITAS
MERCU BUANA



(Wawan Gurawan, S.Kom, MT)
Koord. Tugas Akhir Teknik Informatika



(Emil R. Kaburuan, Ph.D.)
Ka. Prodi Teknik Informatika

PREFACE

Praise be given to God Almighty who has given His grace and guidance, to complete this thesis report for completing the Bachelor's Degree in Informatics Engineering at Universitas Mercu Buana. The author is fully aware in making this thesis report can't be done without any support, thus I'd like to express my gratitude to:

1. Yaya Sudarya Triana, M.Kom., Ph.D. as Dean of Faculty of Computer Science, Universitas Mercu Buana.
2. Emil R. Kaburuan, Ph.D., as Head of Department of Informatics Engineering, Universitas Mercu Buana.
3. Dr. Ruci Meiyanti, M.Kom, as Secretary of International Department of Informatics Engineering, Universitas Mercu Buana.
4. Prastika Indriyanti S.Kom, M.Cs as Academic Supervisor.
5. Dr. Leonard Goeirmanto as Thesis Supervisor.
6. All Lecturers have been dedicated to transferring the knowledge.
7. Both parents always provide prayers and support.
8. Informatics Engineering Classmates who have always been together since the first time starting college at Universitas Mercu Buana.

The author realizes that there are still many shortcomings in this report. Therefore, constructive feedback is allowed and I hope this report will be of use to other parties someday.

Jakarta, 22 February 2022



Robby Auziqni

TABLE OF CONTENTS

TITLE PAGE	i
ORIGINALITY STATEMENT SHEET	ii
THESIS COPYRIGHT LICENSE.....	iii
THESIS PUBLICATION STATEMENT.....	iv
APPROVAL SHEET	v
EXAMINER APPROVAL SHEET.....	vi
COMMITTEE APPROVAL SHEET	vii
ABSTRAK	viii
ABSTRACT.....	ix
PREFACE.....	x
TABLE OF CONTENTS	xi
JOURNAL PAPER.....	1
WORKING PAPER.....	11
CHAPTER 1. LITERATURE REVIEW.....	12
CHAPTER 2. ANALYSIS AND DESIGN.....	21
2.1. Data Collection.....	22
2.2. Preprocessing.....	23
2.3. Model Training.....	23
2.4. Evaluation.....	26
CHAPTER 3. SOURCE CODES.....	28
3.1. Environments.....	28
3.1.1 Software Specifications.....	28
3.1.2 Hardware Specifications.....	28
3.2. Libraries Used.....	29
3.2.1 OpenCV.....	29
3.2.2 MediaPipe.....	29
3.2.3 TensorFlow.....	29
3.2.4 Sci-kit Learn.....	30
3.2.5. Matplotlib.....	30
3.2.6. Seaborn.....	30
3.2.7. NumPy.....	30
3.2.8. Pandas.....	31

3.3. Source Codes of Sign Language Recognition System.....	31
3.3.1. Preparation.....	31
3.3.2. Data Collection.....	33
3.3.3 Preprocessing.....	35
3.3.4 Build and Train the model.....	36
3.3.5 Evaluation.....	37
CHAPTER 4. DATASETS.....	46
CHAPTER 5. EXPERIMENTS	53
CHAPTER 6. RESULTS.....	55
BIBLIOGRAPHY.....	62
ATTACHMENT OF INTELLECTUAL PROPERTY RIGHTS	
DOCUMENTS (HAKI).....	65
1. Dokumen Pernyataan HAKI.....	65
2. Hasil Scan Foto Copy KTP Berwarna.....	67
ATTACHMENT OF CORRESPONDENCE.....	68
CURRICULUM VITAE.....	70



BISINDO Indonesian Sign Language Recognition Using MediaPipe Holistic and LSTM Deep Learning Model

1st Robby Auziqni
Universitas Mercu Buana
Jakarta, Indonesia
41518010085@student.mercubuana.ac.id

2nd Leonard Goeirmanto
Universitas Mercu Buana
Jakarta, Indonesia
leonard@mercubuana.ac.id

Abstract— There is a gap between normal people and deaf people regarding communication. This is because not all normal people learn sign language like deaf people. Therefore, there needs to be something that can close the gap by utilizing technology that can recognize sign language. There are two sign languages used widely in Indonesia, i.e., *Sistem Isyarat Bahasa Indonesia (SIBI)* and *Bahasa Isyarat Indonesia (BISINDO)*. The one that is used as the official sign language at school is *SIBI*, while *BISINDO*, on the other hand, is the one more commonly used by the deaf. In this research, the author presents their findings to recognize *BISINDO* Indonesian sign language from a series of gestures. The author collects a total of 780 image sequences for 26 *BISINDO* alphabet signs. The author utilizes MediaPipe holistic to extract landmarks of the face, hands, and body in each frame of an image sequence to then be processed with Long Short Term Memory (LSTM) deep learning model. The results show that the best model in this research gets an accuracy of 97%, concluding that the model produces decent results in recognition of *BISINDO* sign language alphabet gestures.

Keywords— *BISINDO, Sign Language Recognition, MediaPipe holistic, LSTM, Deep Learning.*

I. INTRODUCTION

There is a gap between normal people and deaf people regarding communication. This is because not all normal people learn sign language like deaf people. Therefore, there needs to be something that can close the gap by utilizing technology that can recognize sign language. There are two sign languages used widely in Indonesia, i.e., *Sistem Isyarat Bahasa Indonesia (SIBI)* and *Bahasa Isyarat Indonesia (BISINDO)*. The one that is used as the official sign language at school is *SIBI*, while *BISINDO*, on the other hand, is the one that is more commonly used by the deaf on their daily life. *SIBI* turns Indonesian spoken language into

sign language; hence it follows the grammatical structure like prefix and suffix. *BISINDO*, on the other hand, express a word from Indonesian spoken language according to the context of the topic. The use of *SIBI* in everyday communication is not fully accepted and used by the Deaf community. The application of Indonesian sentence grammar, such as the application of affixes makes them difficult to communicate. *BISINDO* is a language cues learned naturally by the deaf community. Its speed and practicality make the deaf community easier to understand [1]. Computer vision has been expanded into a vast area of artificial intelligence. The tasks in computer vision are mostly related to the process of obtaining information on events or descriptions, from input scenes (digital images) and feature extraction [2]. With the use of Computer Vision, one should be able to create a Sign Language Recognition system.

II. RELATED WORKS

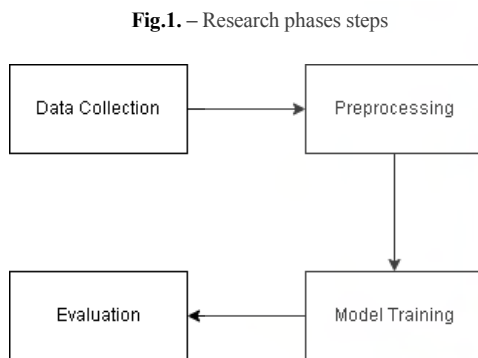
Researchers have become interested in the challenge of developing a reliable model for sign language recognition this decade. This is demonstrated by the numerous studies carried out such as in [3]–[15] to determine the most effective method to handle this situation, including the use of a method similar to the one that will be used in this study, namely using MediaPipe as the primary utility to extract keypoints from the face, pose, and both hands, and then processing it with a deep learning model as has been done in research [3]–[7], and [9]. For example, in research [3], the author uses MediaPipe with Long Short-Term Memory (LSTM) model to detect Indian Sign Language (ISL) and got an accuracy of 90%. In [7] the author did the same thing, only with American Sign Language (ASL) this time, and got an average of 94% accuracy. Similarly, [6] uses 3 different LSTM models to detect Thai Sign Language and got an average of 95%

accuracy. On the other hand, [4] uses CNN and LSTM and got the best out of the two with 98.8% accuracy.

Additionally, other research studies employ slightly distinct methodologies from the study already stated, such as the use of the Hidden Markov Model (HMM) in [10] which achieved 60% accuracy, while [14] uses Leap Motion Controller (LMC) to collect 3D hand skeletal video data of ASL, Processed with Bidirectional Recurrent Neural Network Bi-RNN and LSTM models and got the average of 97% accuracy.

III. METHODOLOGY

This research focused on the implementation of Computer Vision techniques utilizing python libraries such as MediaPipe holistic to extract keypoints / landmarks from frames in an image sequence to then be processed into LSTM Deep Learning architecture, producing a decent model to be then used to Recognize *BISINDO* Indonesian Sign Language gestures. The research is divided into phases as depicted in Fig.1.



A. Data Collection

The data collection phase was carried out independently by the author. The purpose of this phase is to collect data for 26 *BISINDO* alphabet gestures that will be processed in this study. This phase consists of several steps, namely setting up folders, recording videos by utilizing a webcam device in aid with OpenCV, Extracting keypoints with MediaPipe Holistic, and saving the data into .npy files with NumPy.

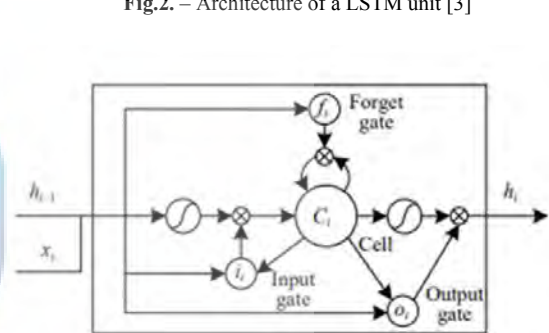
B. Preprocessing

After the data collection phase, the next phase is preprocessing. This phase aims so that the data can be processed by machine learning algorithms for the next phase, namely Model Training. This phase broadly includes merging the data files that have been collected in the previous phase, converting them into a suitable format, then dividing them into 3 parts, namely training data, test data, and validation data. Each data will have dimensions of 780x30x1662, with 780 as the total number of videos, 30 being the number of frames per video, and 1662 being the number of keypoints obtained from the extraction with MediaPipe Holistic.

C. Model Training

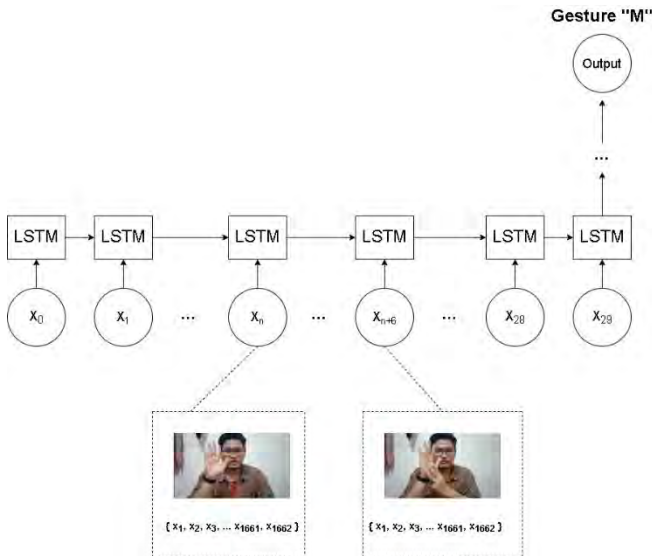
In this phase, several deep learning models are built and trained. The architecture of the models is similar to what has been used in past research such as in [3] and [9]. The use of Long Short Term Memory (LSTM) became emphasized in this research. Long Short Term Memory (LSTM) is a type of Recurrent Neural Network (RNN). It was designed by Hochreiter & Schmidhuber, an RNN-specific architecture created for sequential data prediction that can analyze time-series data over extensive time periods and automatically choose the best time-lags [16], [17]. It addressed the issue of long-term dependencies of RNN in which the RNN cannot predict the information stored in the long-term memory but can give more accurate predictions from the recent information. RNN does not operate effectively as the gap length rises. LSTM can by default retain the information for a long period of time. It is utilized for processing, predicting, and classifying on the basis of time-series data. Therefore, it is suitable to process image sequences or video data.

Fig.2. – Architecture of a LSTM unit [3]



A single LSTM model's architecture can be described as having three gates and a component known as a memory cell. Information that is no longer useful is removed from the cell state by the forget gate. The forget gate specifies the types of information that can be forgotten and are no longer relevant in the context. The input gate enhances the cell state with crucial information. It addresses what fresh information we ought to include or update in our operational storage state data. The output gate is in charge of retrieving valuable information from the current cell state and presenting it as output. The memory cell has two states: a self-state known as long-term memory, and short-term memory. Based on how each gate functions, values between 0 and 1 are assigned to each gate. The information passed depends on the value; if it is 0, no information is passed; if it is 1, all information is passed; and also, on the values in between [3].

Fig.3. – Prediction or matching process



The way this model works is from how the LSTM detects a similar pattern from the available data, by deciding to store important information in a frame in the form of 1662 keypoints through the gates contained in each unit and combining the information to produce an output, namely the hidden state which will be processed in the next layer and then predict a label according to the information obtained during the training process. For example as depicted in Figure 5, the LSTM decides that the information contained in frame x_n is important, this information will be stored and then be forwarded to the next LSTM unit, until frame x_{n+6} , there is other important information, the LSTM decides that the previous information is still relevant, then these two pieces of information will then be processed by the next unit to find other important information, then generate a hidden state which will be processed by the next layer and generate gesture prediction, in this case, gesture of alphabet "M".

The model in this research mainly utilizes two types of layers, namely the LSTM layers and Dense layers. The author built three models with the same architecture consisting of 6 layers, 3 LSTM, and 3 Dense layers. The first 5 layers use 'ReLU' activation function, while the last one uses 'softmax'. After this phase, the performance of the three models is then compared and evaluated in the next phase.

D. Evaluation

After all deep learning models are trained, the next step is to evaluate these models. Evaluation is done by looking at the graph of the accuracy and loss of the training process of each model using a visualization tool from TensorFlow called Tensorboard. Accuracy and loss of training and validation will be seen and compared between models. After that, other parameters used for evaluation are the confusion matrix, the accuracy score, and the f1 score.

Confusion Matrix is one of the performance indicators used to evaluate machine learning classification problems where the output can be two or more classes. Recall, Precision, Specificity, Accuracy, and most significantly AUC-ROC curves are all measurements that may be made with great benefit from it [18]

Fig.4. – A confusion matrix [18]

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Confusion Matrix is a table of different combinations of predicted and actual values, namely TP, TN, FP, and FN.

- TP (True Positive): The model predicted positive and it's true.
- TN (True Negative): The model predicted negative and it's true.
- FP (False Positive): The model predicted positive and it's false.
- FN (False Negative): The model predicted negative and it's false.

From these values, one can get other derived measurements such as Recall, Precision, Accuracy, and F-measure.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Recall = \frac{TP}{TP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

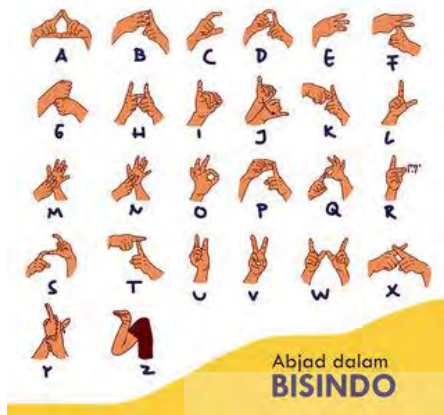
$$F1 - score = \frac{2 * Recall * Precision}{Recall + Precision}$$

- Recall quantifies the number of positive class predictions made out of all positive examples in the dataset.
- Precision measures the proportion of predictions from the positive class that actually fall into it.
- Accuracy is the total amount of correct prediction from all classes.
- F-measure, also called F1-score, gives a single score that balances the concerns of recall and precision in a single value.

IV. EXPERIMENT AND RESULTS

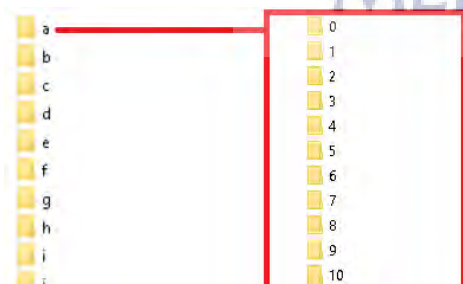
As previously mentioned, the purpose of this research is to create a model that can recognize *BISINDO* Indonesian Sign Language alphabet gestures. The gestures used in this research can be seen in Fig.5.

Fig.5. – *BISINDO* Alphabet gestures [19]



the dataset used in this study undergoes several forms of adjustment through the data collection and preprocessing phase. In the data collection phase, the first step is to set up folders for data collection using a program written in Python. The program will create 26 folders regarding the gestures that will be collected (*BISINDO* Alphabet gestures A-Z), then in each of them there are 30 folders (named 0-29), because the amount of videos / image sequences that will be collected for each gesture is 30.

Fig.6. – Folder setup



The next step is to collect the data by using a webcam device. This was done by using the Python library OpenCV in aid with other libraries such as NumPy and MediaPipe. Due to the amount of expected data for each gesture being 30, there will be 780 videos in total. In each video, there will be 30 frames where the signer will perform the gesture. MediaPipe Holistic will then be utilized to detect the signer, draw the landmarks, and extract the total of keypoints of face, pose, and both hands from the landmarks for each frame.

Fig.7. – Pose landmarks [20]

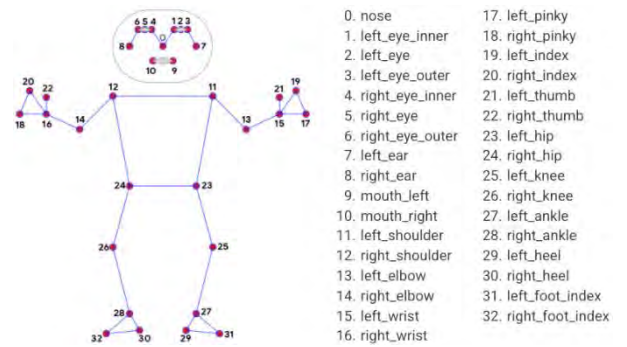


Fig.8. – Hand landmarks [21]



MediaPipe Holistic pipeline combines separate models for the pose, face, and hand components, each of which is optimized for its particular domain [22]. The landmark model in MediaPipe Pose predicts the location of 33 pose landmarks [20], while MediaPipe Face Mesh predicts 468 3D face landmarks [23]. Similarly, MediaPipe Hands predicts the location of 21 3D landmarks for each hand [21]. This makes the total of 543 landmarks being predicted by MediaPipe Holistic. Each landmark in Pose, Face, and Hands model consists of keypoints containing three coordinates, which are x, y, and z while the MediaPipe Holistic Pose has an additional keypoint, which is visibility. This makes the total of keypoints in 543 landmarks being 1662 keypoints, 1404 in Face, 132 in Pose, and 126 in Hands respectively. The extracted keypoints then will be converted into NumPy arrays, concatenated into one array, then saved into its corresponding folder as a .npy file as depicted in Fig.12.

Fig.9 – Detect signer and draw landmarks

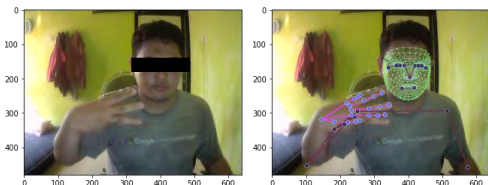


Fig.10. – Landmark data

```
Out[52]: landmark {
  x: 0.6335775852203369
  y: 0.40839359164237976
  z: -0.9923591613769531
  visibility: 0.9993411898612976
}
landmark {
  x: 0.6576088200645447
  y: 0.34050142765045166
  z: -0.9489122629165649
  visibility: 0.9991867542266846
}
landmark {
  x: 0.6729504466056824
  y: 0.34063392877578735
  z: -0.9490703344345093
  visibility: 0.9991664886474609
}
landmark {
```

Fig.11. – Extracted keypoints

```
Out[50]: array([ 0.63357759, 0.40839359, -0.99235916, ..., 0.40825266,
 0.5181855 , -0.04047192])
```

Next, in the preprocessing phase, the data is merged into one multi-dimensional NumPy array with the shape of (780, 30, 1662). This array holds the independent variable X, which then be mapped to another NumPy array consisting of labels that will become the dependent variable y. With the help of the Keras library, the y then is converted into a binary array, e.g. [1, 0, 0, ... 0, 0, 0] with 1 indicating the label, as depicted in Fig. 13.

Fig.12. – Save the keypoints as npy files

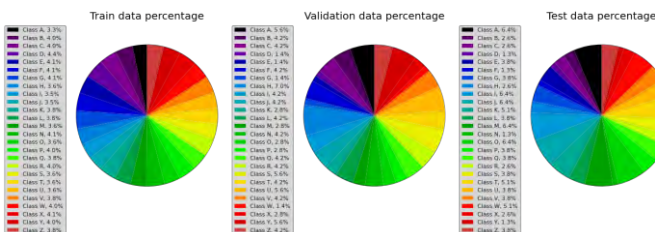
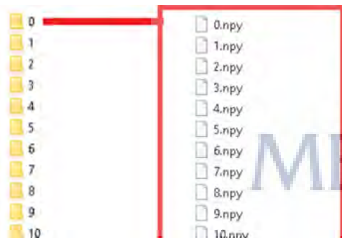


Fig.13. – labels (y) dataset, before (left) and after(right) transformation

After we get the array of X and y, the next step is to split

alphabet_index		a	b	c	d	e	f	g	h	i	j	...	q	r	s	t	u	v	w	x	y	z	
0	0	0	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0	0
2	0	2	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0	0
3	0	3	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0	0
4	0	4	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0	0
...
775	25	775	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0	1
776	25	776	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0	1
777	25	777	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0	1
778	25	778	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0	1
779	25	779	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0	1

780 rows x 1 columns 780 rows x 26 columns

them into train, test, and validation data. This was done with the help of the Scikit-Learn library. The ratio of the split for the train, test, and validation data is 8:1:1 respectively. This means the training dataset will have about 631 data, while the test dataset will have about 78 data, and the validation dataset will have about 71 data, for more detail about the information on the amount of data and percentage for each class in each dataset, please refer to Fig. 14. and Fig. 15. Next, the data is ready to be processed into Deep Learning model.

Fig.14. – The amount of data for each class in each dataset

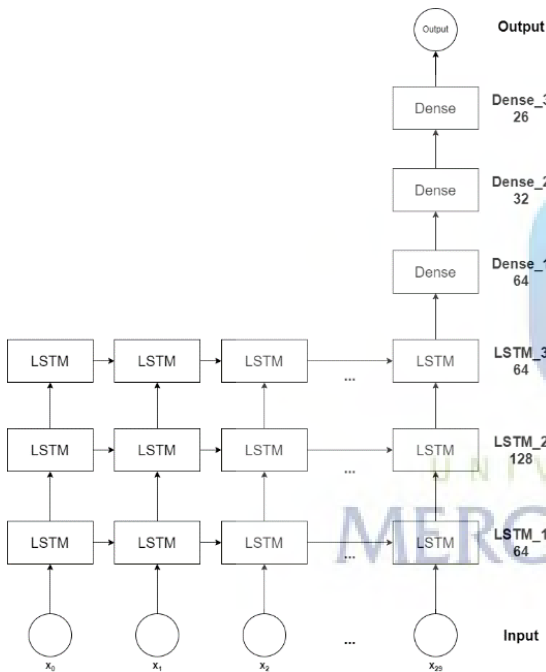
	Train	Val	Test
a	21	4	5
b	25	3	2
c	25	3	2
d	28	1	1
e	26	1	3
f	26	3	1
g	26	1	3
h	23	5	2
i	22	3	5
j	22	3	5
k	24	2	4
l	24	3	3
m	23	2	5
n	26	3	1
o	23	2	5
p	25	2	3
q	24	3	3
r	25	3	2
s	23	4	3
t	23	3	4
u	23	4	3
v	24	3	3
w	25	1	4
x	26	2	2
y	25	4	1
z	24	3	3

Fig.15. – Class percentages in each dataset

The next step is the model training phase. The author has experimented with the architecture of the model before deciding to use a particular architecture in this study. Some

of these experiments are adding and reducing the number of LSTM layers, and changing the shape of the layers. For the final decision, as have been mentioned earlier, the author uses deep learning model architecture consisting of 6 layers, namely 3 LSTM layers and 3 Dense layers such as depicted in Fig.16. The first 5 layers use the 'ReLU' activation function, while the last layer uses 'Softmax'. Furthermore, in this research, the author builds three models with the same architecture to compare their performance. The difference between these three models lies in the different epoch parameters when the model starts training. The epochs used in the experiment are 100, 250, and 500 to compare how well these three models perform with the same data. The optimizer used is ADAM to deal with stochastic gradient descent. During the training of the models, both training and validation accuracy and loss are measured after each epoch for evaluation.

Fig.16. – Model's Architecture Diagram



The outcomes of the trained models need to be assessed after experimentation. The accuracy and loss performance of the models after training are the first things to be assessed.

Fig.17. – Model 1 Training performance

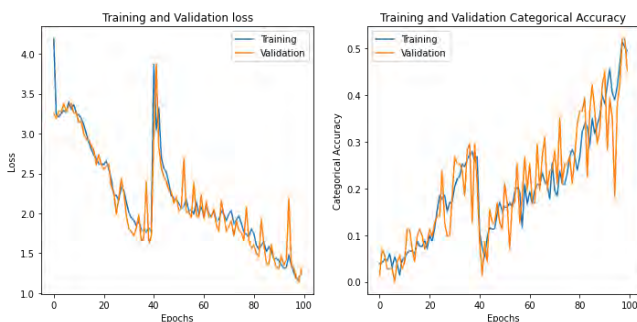


Fig.18. – Model 2 Training performance

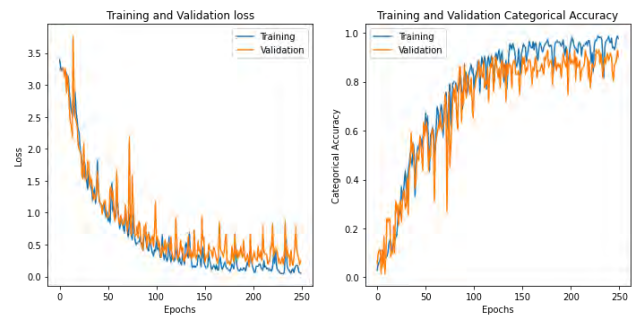
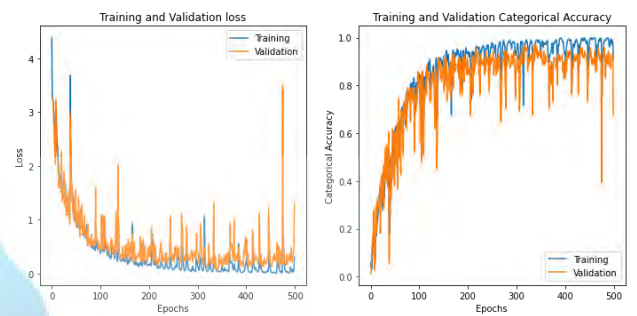


Fig.19. – Model 3 Training performance



We begin by examining the performance of model 1, which is trained using just 100 epochs, as was stated in the preceding chapter. As seen in Fig.17, the loss and accuracy training graph for this model is quite shaky; train data and validation data frequently do not coincide, which is a sign of subpar performance. The utmost accuracy that this model can obtain during training, particularly when in epoch 90 and above, only reaches 50%, serves as more evidence of this.

While this is happening, in model 2, the accuracy and loss graphs appear to be fairly stable, the accuracy graphs appear to be gradually increasing while the loss graphs appear to be gradually decreasing. In addition, the performance of the training data and validation data appears to be closely correlated, and the values are not too dissimilar. Because it does not indicate any overfitting, this suggests that the model is certainly a decent one. The maximum accuracy that this model can reach on epoch 250 is roughly 99%, while the loss is approximately 0,3%.

Model 3 on the other hand, is somewhat like model 2 and model 1, where the accuracy rises and the loss graph gradually decreases, while the performance of this model is also occasionally unstable, and there are numerous instances where the performance of the training data and the validation data deviates significantly from one another. Particularly during epochs of 450 and above. This could be an indication of overfitting, when the model continuously tries to learn what has been learned after a large number of epochs, ultimately leading to a significant disparity between the performance of the training data and the validation data. Ultimately, in epoch 500, the loss increased

to around 1.5% while the accuracy decreased to almost 60%. From these observations, it can be concluded that the best model of these three experiments is model 2, the model with 250 epochs.

Fig.20. – Multiclass Confusion Matrix of Model 1

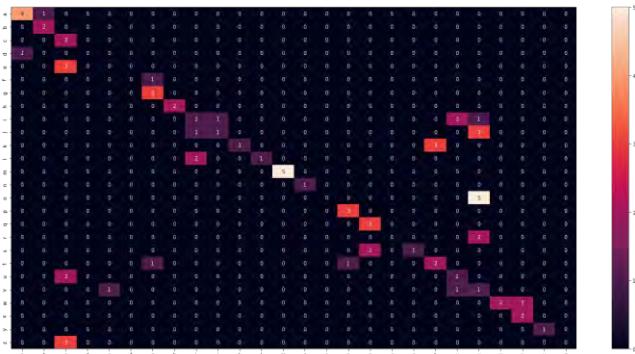


Fig.21. – Multiclass Confusion Matrix of Model 2

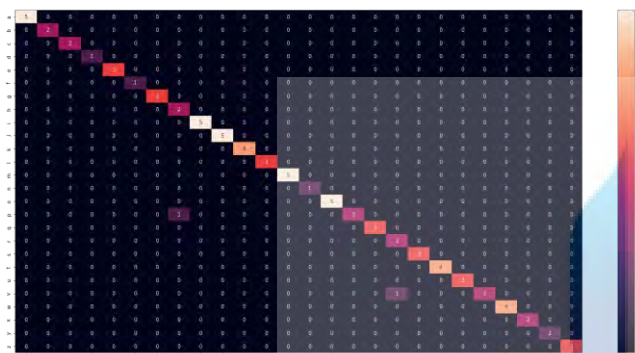
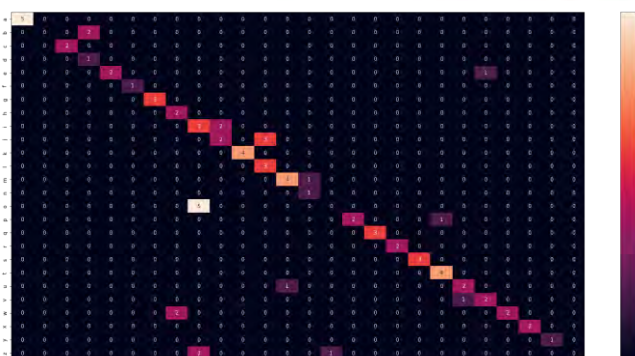


Fig.22. – Multiclass Confusion Matrix of Model 3



In addition to performance during training, the author also tries to evaluate the performance results of these models with a confusion matrix based on test data. Model 1 contains a lot of incorrect predictions based on the general number of False Positive and False Negative instances. Class "V" had the most False Positive cases, with a total of

11, compared to just 1 True Positive. Additionally, many classes have TP values of 0, which means that this model is either unable to detect those classes or will undoubtedly be incorrect in its predictions of those classes. Furthermore, as depicted in Fig.20., the author tries to uncover the specifics of this model's poor performance. We can see that this model misidentified "I," "J," "O," and "R" as "V" 1, 3, 5, and 2 times, respectively.

If we compare model 2 to model 1, there are essentially no instances of missed predictions from this model, as shown by the Confusion matrix in Fig.21., only 2 instances of predictions from this model are incorrect, specifically 1 False Negative in class "P" and "V" this was because the model incorrectly predicts 'V' as 'R' and 'P' as 'H'.

Moving on to model 3, referring to Fig.22., here we can see that similar to model 1, there are many errors in model 3, although not as many as in model 1. The most prominent is the case of 7 False Positive in class 'I', and 5 cases of False Negative in class 'O'. This is because this model incorrectly predicts 'O' and 'Z' as 'I', 5 and 2 times, respectively, and predicts 'I' when in fact it is 'O', which is unique in that the two cases intersect with each other. Now that we have the confusion matrices for the three models, we can calculate other performance metrics, including test accuracy and f1 score. With the help of Sci-kit learn, the author can produce results from the three models as depicted in TABLE I below.

TABLE I. PERFORMANCE COMPARISON OF THE THREE MODELS

Model	Precision	Recall	Accuracy	F1-Score
Model 1	0.52	0.53	0.50	0.47
Model 2	0.97	0.97	0.97	0.96
Model 3	0.71	0.76	0.71	0.76

In keeping with the preceding observation, it can also be deduced and proven from this observation that model 2 is the best out of the three, followed by model 2 and finally model 1.

TABLE II. PERFORMANCE COMPARISON TO RELATED RESEARCH

Research	Method	Best Model Accuracy
This Research	MediaPipe Holistic + LSTM	97%
[3]	MediaPipe Holistic + LSTM	90%
[4]	MediaPipe Holistic + CNN	97%
[5]	MediaPipe Holistic + RNN	90%
[6]	MediaPipe Holistic + LSTM	97%
[7]	MediaPipe Holistic + LSTM	94%
[8]	E-CNN	99%
[9]	MediaPipe Holistic + Bi-LSTM	92%
[10]	Microsoft Kinect Xbox + HMM	60%
[11]	G-CNN	99%
[12]	Inception v3 CNN	90%
[13]	DenseNet 201	86%
[14]	LMC Sensor + LSTM	97%
[15]	LSTM	97%

The performance score obtained from the best model in this study, which is model 2, when compared with other studies, namely research [3]–[9], [11], [12], [14], [15] is on par. The accuracy value achieved from model 2 in this study, which is 97 %, is fairly comparable with the accuracy value gained from these research studies since these research studies receive an average accuracy value of 90 % or higher from their best models. Additionally, it is unmistakably thought to be superior when compared to research [10], [13] because these studies have an average accuracy of around 75%.

However, when it comes to quality, the models from the research that have been mentioned earlier, are not necessarily bad. In terms of functionality, they may be better than this study, considering the small amount of data used in this study when compared to those studies. Furthermore, for this reason, the best model in this study may not be reliable enough to be used in real-time detection cases, because the more data used, the more variations in the data, and the more variations in the data, the greater the probability of the model to be able to detect data that it has never encountered, which in the case of real-time detection is very likely to happen. For this reason, the authors do not necessarily claim that the technique used in this study is better than other studies. The author hopes that the techniques used in this study can be considered and further developed for future research.

CONCLUSIONS

In this study, an experiment was conducted to create a model that can recognize *BISINDO* sign language. This research focuses on utilizing the MediaPipe Holistic utility to extract keypoints on the face, poses, and hands, then processing them with the LSTM model. From the results of experiments with the three models, it can be concluded that model 2, namely the model with training epoch 250, is the best model among the three, achieving an accuracy of 97% and an f1-score of 96%. These results certainly include commensurate results when compared with the results achieved by other researchers from the studies that have been mentioned. However, this model is still far from perfect, this model is not reliable enough to perform real-time detection considering the small amount of data used. Therefore, the authors hope that in future research that uses a similar method, please add more data so that the resulting model can be much more perfect.

REFERENCES

- [1] G. Gumelar, H. Hafiar, and P. Subekti, "KONSTRUKSI MAKNA BISINDO SEBAGAI BUDAYA TULI BAGI ANGGOTA GERKATIN," *INFORMASI*, vol. 48, no. 1, p. 65, Jul. 2018, doi: 10.21831/informasi.v48i1.17727.
- [2] V. Wiley and T. Lucas, "Computer Vision and Image Processing: A Paper Review," *International Journal of Artificial Intelligence Research*, vol. 2, no. 1, p. 22, Jun. 2018, doi: 10.29099/ijair.v2i1.42.
- [3] A. S. Agrawal, A. Chakraborty, and C. M. Rajalakshmi, "International Journal of Research Publication and Reviews Real-Time Hand Gesture Recognition System Using MediaPipe and LSTM," *International Journal of Research Publication and Reviews*, vol. 3, pp. 2509–2515, 2022, [Online]. Available: www.ijrpr.com
- [4] S. Dhulipala, F. Adedoyin, and A. Bruno, "Sign and Human Action Detection Using Deep Learning," *J Imaging*, vol. 8, no. 7, p. 192, 2022, doi: 10.3390/jimaging8070192.
- [5] B. Duy Khuat, D. Thai Phung, H. Thi Thu Pham, A. Ngoc Bui, and S. Tung Ngo, "Vietnamese sign language detection using Mediapipe," in *ACM International Conference Proceeding Series*, Feb. 2021, pp. 162–165. doi: 10.1145/3457784.3457810.
- [6] A. Chaikaew, K. Somkuan, and T. Yuyen, "Thai Sign Language Recognition: An Application of Deep Neural Network," in *2021 Joint 6th International Conference on Digital Arts, Media and Technology with 4th ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering, ECTI DAMT and NCON 2021*, Mar. 2021, pp. 128–131. doi: 10.1109/ECTIDAMTNCNCON51128.2021.9425711.
- [7] A. Domènech, "SIGN LANGUAGE RECOGNITION ASL Recognition with MediaPipe and Recurrent Neural Networks," Bachelor Thesis, UPC, Barcelona, pp. 1-40, 2020. [Online]. Available: <http://hdl.handle.net/2117/343984>
- [8] C. Suardi, A. N. Handayani, R. A. Asmara, A. P. Wibawa, L. N. Hayati, and H. Azis, "Design of Sign Language Recognition Using E-CNN," in *3rd 2021 East Indonesia Conference on Computer and Information Technology, EICoNCIT 2021*, Apr. 2021, pp. 166–170. doi: 10.1109/EICoNCIT50028.2021.9431877.
- [9] H. Moetia Putri and W. Fuadi, "PENDETEKSIAN BAHASA ISYARAT INDONESIA SECARA REAL-TIME MENGGUNAKAN LONG SHORT-TERM MEMORY (LSTM)," *TTS 4.0*, vol. 3, no. 1, pp. 14–25, 2022, [Online]. Available: <https://ojs.unimal.ac.id/tts/article/view/6853>
- [10] T. Handhika, R. I. M. Zen, Murni, D. P. Lestari, and I. Sari, "Gesture recognition for Indonesian Sign Language (BISINDO)," in *Journal of Physics: Conference Series*, Jun. 2018, vol. 1028, no. 1. doi: 10.1088/1742-6596/1028/1/012173.
- [11] S. Sharma and S. Singh, "Vision-based hand gesture recognition using deep learning for the interpretation of sign language," *Expert Syst Appl*,

- vol. 182, Nov. 2021, doi: 10.1016/j.eswa.2021.115657.
- [12] Ahmed Elhagry and Rawan Gla Elrayes, "Egyptian Sign Language Recognition Using CNN and LSTM," *CoRR*, vol. abs/2107.13647, pp. 1–7, 2021, [Online]. Available: <https://arxiv.org/abs/2107.13647>
- [13] N. H. Phong and B. Ribeiro, "Action Recognition for American Sign Language," *ArXiv*, vol. abs/2205.12261, pp. 1–2, May 2022, [Online]. Available: <http://arxiv.org/abs/2205.12261>
- [14] 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.
- [15] D. Kothadiya, C. Bhatt, K. Sapariya, K. Patel, A.-B. Gil-González, and J. M. Corchado, "Deepsign: Sign Language Detection and Recognition Using Deep Learning," *Electronics (Basel)*, vol. 11, no. 11, p. 1780, Jun. 2022, doi: 10.3390/electronics11111780.
- [16] I. Nurhaida *et al.*, "Implementation of Deep Learning Predictor (LSTM) Algorithm for Human Mobility Prediction," *International Journal of Interactive Mobile Technologies*, vol. 14, no. 18, pp. 132–144, 2020, doi: 10.3991/ijim.v14i18.16867.
- [17] S. Hochreiter and J. "Urgen Schmidhuber, "Long Short-Term Memory," *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.
- [18] Sarang Narkhede, "Understanding Confusion Matrix," *Towards Data Science*, May 09, 2018. <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62> (accessed Jun. 20, 2022).
- [19] Klobility, "Klobility - BISINDO dan SIBI: Apa Bedanya?," *klobility*, Sep. 23, 2019. <https://www.klobility.id/post/perbedaan-bisindo-dan-sibi> (accessed Jun. 20, 2022).
- [20] MediaPipe, "Pose," *MediaPipe*, 2020. <https://google.github.io/mediapipe/solutions/pose.html> (accessed Jun. 20, 2022).
- [21] MediaPipe, "Hands," *MediaPipe*, 2020. <https://google.github.io/mediapipe/solutions/hands.html> (accessed Jun. 20, 2022).
- [22] MediaPipe, "Holistic," *MediaPipe*, 2020. <https://google.github.io/mediapipe/solutions/holistic> (accessed Jun. 20, 2022).
- [23] MediaPipe, "Face Mesh," *MediaPipe*, 2020. https://google.github.io/mediapipe/solutions/face_mesh.html (accessed Jun. 20, 2022).



UNIVERSITAS
MERCU BUANA



WORKING PAPER

Summary

The working paper is a complete material for a journal titled “BISINDO Indonesian Sign Language Recognition Using MediaPipe Holistic and LSTM Deep Learning Model”. The working paper contains all the material for the results of the Journal that are not included in the Journal. This paper presents several sections which consist of literature review, analysis and design, source code, dataset, experiment stage, and results of all the experiments.

Chapter 1 discusses the literature review, which includes journal articles that serve as the base or cornerstone of this research. Chapter 2 explains the research's analysis and design. A flowchart is used to demonstrate the design in this chapter. In Chapter 3, the source code that was utilized in this study is described, along with an explanation of how each code was applied to the processing of the data at hand. The dataset utilized in this study is described in detail in Chapter 4, along with information on how to collect the data, its properties, and any modifications needed to make the final data processing-ready. Chapter 5 explains the author's progressive experimentation, backstage processes the author went through during this research. It describes the stages that must be passed in processing the dataset into the expected research results. The final section of this research, Chapter 6, describes the general findings of the experiments that have been conducted, including the calculation of performance results using the Confusion Matrix and F1-Score.