Finger Recognition Detection System Using Mediapipe as Communication Solution for People with disabilities

M.Wahyu Elfander and Ryan Satria Wijaya*

Politeknik Negeri Batam, Batam, Riau Island 29461, Indonesia
ryan@polibatam.ac.id

Abstract. Sign language is a language used by people with disabilities, especially the deaf and mute to communicate. The problem is, not everyone can understand sign language. This study aims to create a system that can translate sign language. The system can also produce sound with the text to voice method. The system is built using Mediapipe to detect fingers that form sign language. The system performs classification by combining 2 machine learning models with the Artificial Neural Network (ANN) method. The first model is used to classify letters A-Z and the second model is used to classify movement patterns in letters J and Z. The accuracy of the first model is 94% and the accuracy of the second model is 95%. The model will also be evaluated using the confusion matrix technique to find recall, f1-score, and precision of 25 letters or classes.

Keywords: Sign Language, Confusion Matrix, Artificial Neural Network

1 Introduction

People with disabilities is a term used to refer to individuals who have physical, mental, sensory or visual limitations that affect their daily activities [1]. People who cannot speak are called mute and people who cannot hear are called deaf. Mute and deaf people can also be called disabled people [2]. This journal will more specifically discuss people with speech or hearing disabilities. People who are mute can only use sign language as a language of communication to people in general and to fellow mute communities [3]. Likewise with deaf people.

The problem is, not everyone can talk and not everyone understands sign language. This research aims to build a system that can translate sign language. This system will use a camera to detect sign language fingers and then store each sign language letter to form a sentence. The system can also provide voice output based on stored sentences. This system can be used by people with disabilities (mute and deaf) and can also be used by normal people in general.

When this system is used by people with disabilities (mute and deaf), the voice feature can help as a medium for conveying information. When this system is used by normal people in general, the sign language finger detection feature can help normal people who do not understand sign language. The sign language used in this research is SIBI. SIBI is an abbreviation of the “Sistem Isyarat Bahasa Indonesia”. SIBI is

© The Author(s) 2024
L. Lumombo et al. (eds.), *Proceedings of the 7th International Conference on Applied Engineering (ICAE 2024)*,
Advances in Engineering Research 251,
https://doi.org/10.2991/978-94-6463-620-8_11

recommended by the Indonesian government for use because it only uses one hand [4]. SIBI also has similar patterns to ASL (American Sign Language). ASL is adopted by many countries. So ASL is international [5].

2 Method

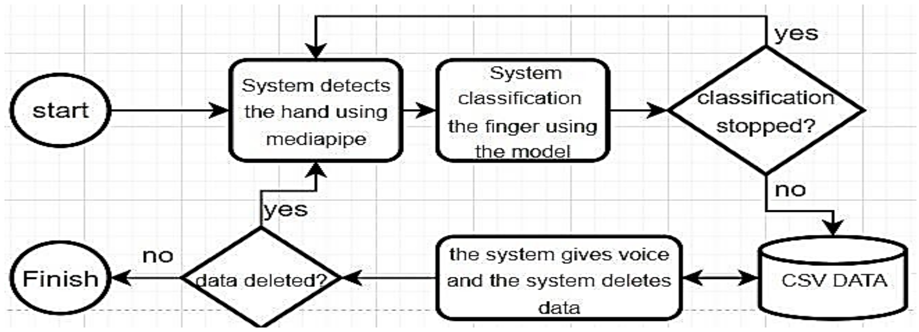

start
System detects the hand using mediapipe
System classification the finger using the model
classification stopped?
yes
no
CSV DATA
the system gives voice and the system deletes data
data deleted?
yes
no
Finish

Fig. 1. System design.

Based on Figure 1, the system built will use Mediapipe to detect hand landmarks. When detecting hand landmarks, the system is also programmed to create a line connecting the landmarks of each finger using the drawing function in OpenCV called line (CV2.line). Then the system will classify sign language based on fingers using 2 models that have been deployed to TFLite [6]. If the system cannot detect each letter based on the sign language fingers for up to 5 seconds, the classification will be canceled and the time will be reset. However, if the system succeeds in classifying for up to 5 seconds, the classification result data will be saved into CSV data. The system can produce sounds based on letters that form 1 or more sentences in the CSV data. For the system to produce sounds, the pyttsx3 library is used [7]. The reason for using pyttsx3 is that it is offline and there are parameters to set the voice accent [8]. In this study, the voice accent used is "Bahasa Indonesia" with a voice speed of 70. The way pyttsx3 works is text to voice conversion. Pyttsx3 will read every sentence contained in the CSV data [9].

2.1 Data Collection

The data collection process will be carried out for the dataset used as the input in first model and seccond model. Both models utilize the Mediapipe library to obtain hand landmark coordinates [10]. Based on Figure 2, there are 21 hand landmarks with a range of 0-20 [11]. In model 1, data retrieval is carried out by taking coordinates based on the shape of the Sign Language finger pattern on letters A to Z [12]. The data retrieval is carried out with 2 hands. To distinguish each letter, each letter will be given a unique value in the dataset called Class_Id [13]. There are 26 Class_Id, with each data having 42 features for model 1. The 42 features are based on taking the X and Y coordinates of each hand landmark. 500 data were taken for each data. Data distribution for both models, 80% for train data and 20% for test data

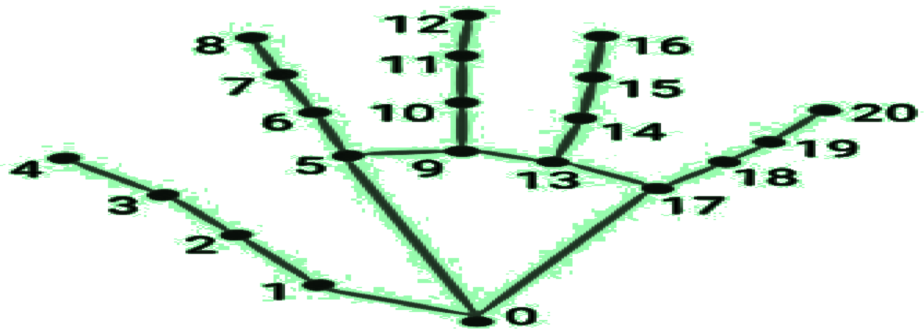


Fig. 2. Hand coordinate points on Mediapipe.

Based on SIBI Sign Language, the letters J and Z require finger movements, so the movement recording is done separately for model 2. Although the dataset of letters J and Z already exists for model 1, model 1 does not support movement. In this system, the letters J and Z will form a circle trail on the fingertips. The drawing of the circle is based on the landmark of the fingertips (8 and 20) on the index finger and little finger. The drawing of the circle uses the OpenCV Library (cv2.circle). If the system detects the letters J and Z, the circle will appear 16 times using a for-loop. Similar to model 1, the circle trail on the letters J and Z is distinguished using Class_Id. The fingertip movement pattern will be followed by these circles, and the coordinates of each circle will be taken as a feature for the model 2 dataset, which consists of 16 X and 16 Y, a total of 32 features for model 2.

2.2 Data Train

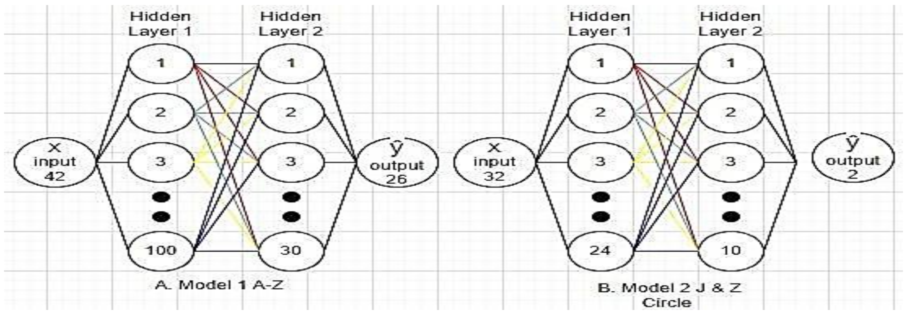


Fig. 3. Model 1 & model 2 architecture.

In the data training process, the collected dataset will be used as input for the model to be built. The purpose of training data on model 1 is to find a similarity pattern for

each Class_Id coordinate [16]. The purpose of training data on model 2 is to find a similarity pattern for each circle coordinate that forms the letters J and Z. In this training process, the method used is to build 2 models using machine learning techniques with the ANN (Artificial Neural Network) algorithm. Both models will be built using the TensorFlow framework.
Based on Figure 3, there are 2 model architectures. Model 1 is a model for classifying letters A-Z based on the coordinates of each previously taken finger landmark. Model 1 has 2 hidden layers with 42 input features. There are 26 classes in model 1. Then model 2 is a model for classifying history steps circle based on the coordinates of the circle at the fingertips of the letters J and Z. In model 2, the number of hidden layers is 2. However, the number of neurons is less than in model 1. The number of input layers in model 2 is 32. The number of classes in model 2 is 2.

$$z_i^l = \sum_j a_j^{(l-1)} W_{ji}^{(l-1)} + b_i^{(l-i)} \tag{1}$$

$$a_i^{(l)} = f^l(z_i^{(l)}) \tag{2}$$

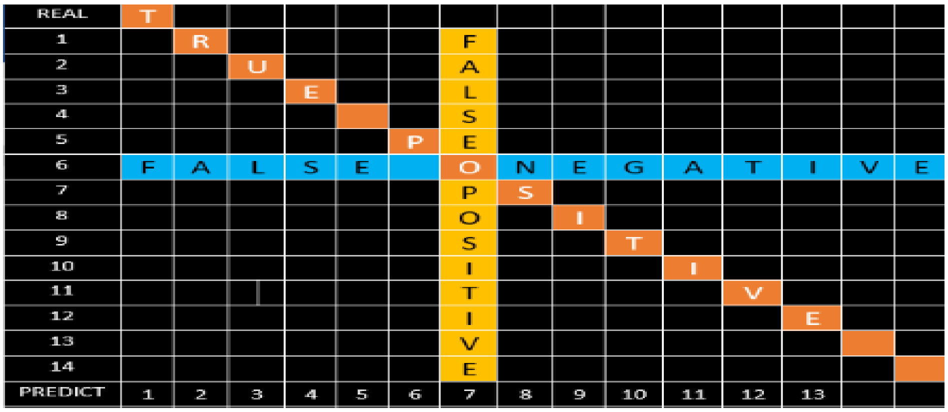
Equation (1) is used in calculating the formula in artificial neural networks. There is a symbol z as input for the activation function [17]. The input for the activation function is obtained from the variable a which is the result of the previous neuron activation function. This variable a is multiplied by the variable W as the weight. Then the result of W is multiplied by a and will be added to b as the bias [18].

Equation (2) shows the activation function used for z. The activation function used for each hidden layer in this study is ReLU [19]. ReLU will output a value of 0 if the value of z is below 0. If z is above 0, then ReLU will give the result of the value itself. In the output layer, the activation function used is softmax. Softmax is commonly used in the output layer for multiclass classification [20].

2.3 Model Evaluation

To calculate model evaluation, the lost function used for both models is sparse categorical crossentropy. Sparse categorical crossentropy is suitable for use in multiclass models [21]. Both models' performance is measured using accuracy matrix. So, in this study, the accuracy value is only obtained from the accuracy matrix in the TensorFlow Keras library. Both models will also be evaluated using confusion matrix.

Confusion matrix is used because confusion matrix can measure classification overlap [22]. For every cell in the matrix, the row represents the actual class and the column represents the class as the classification model produced. As seen in Figure 4, the main diagonal will read True Positive (TP) from each class, TP shows that the model classifies positive from positive test data [23]. The row cells based on the main diagonal will read False Positive (FP) from each class, FP of a class indicates that the class cannot read a small or large part of the class itself. The column cells based on the main diagonal will read False Negative (FN) from each class [24].



REAL	T														
1		R					F								
2			U				A								
3				E			L								
4							S								
5						P	E								
6	F	A	L	S	E		O	N	E	G	A	T	I	V	E
7							P	S							
8							O		I						
9							S			T					
10							I				I				
11							T					V			
12							I						E		
13							V								
14							E								
PREDICT	1	2	3	4	5	6	7	8	9	10	11	12	13		

Fig. 4. Confusion matrix.

FN A class shows that it has misclassified another class. To facilitate the analysis of the confusion matrix, the visualization of the confusion matrix will be displayed in the form of Heat map. In this study, the heat map technique used is the brighter color in the data, the higher the value. In this research, the scikit-learn library will be used to obtain the Confusion matrix from the trained model. The Confusion matrix will provide 7 important reports for each class. The reports are Precision, F1-Score, Recall, Support, TP, FP, and FN. By using the scikit-learn library, the TP, FN, and FP values can be read in the confusion matrix.

So, the confusion matrix not only displays colors, but also the numerical values of TP, FN, and FP. Precision shows the comparison between True Positive (TP) and the number of data that are predicted to be positive in each class [25]. The F1-Score shows the harmonic mean of precision and recall. Recall shows the comparison between True Positive (TP) and the amount of data that is actually positive. Support shows the number of test data samples. In this study, the use of the word "**Support**" is replaced with **test data**. This is because test and support data have the same meaning. In this study also, precision can be calculated using equation (3), recall can be calculated using equation (4), and F1-Score can be calculated using equation (5).

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{3}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{4}$$

$$f1\ Score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{5}$$

3 Result and Implementation

The results and implementation will be done in 3 stages. The first stage is to implement model 1 and 2 to detect sign language A-Z. The second stage is to analyze a little error in the classification results in the first stage. And last stage is the implementation of the system.

3.1 Implementation Model 1 and Model 2 to Detect Sign Language A-Z (First Stage)

In the implementation of model 1, the accuracy is obtained from the evaluation of the trained model 1 is 94%. The results of testing model 1 using Sign Language were successful. Model 1 can classify as many as 25 letters. Evidence of letter classification testing can be seen in Figure 5. However, there are 2 letters that need to be analyzed. The letters are F and U. These two letters are rather difficult to classify. When trying to form a hand with the letter F the system sometimes reads the letter F as the letter B. Likewise with the letter U which is sometimes read as letter R.

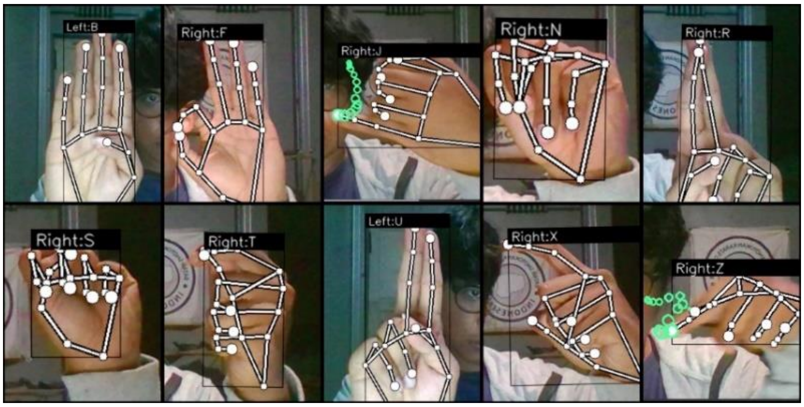


Fig. 5. Letter classification results

In the implementation of Model 2, model 2 also has good accuracy, which is 95%. In Figure 5, for the letters J and Z, the model implementation is good. This is because when the system detects the fingertips of the letters J and Z, model 2 can classify by issuing a green circle. The results of model 2 also illustrate that this green circle can perfectly follow the movement of the letters J and Z. Because model 2 can be implemented well, the focus of the analysis will only be carried out on model 1. Because in model 1 there are 2 letters that are difficult to classify.

3.2 Classification Error Analysis Based on Confusion Matrix Model 1 (Seccond Stage)

Based on Figure 6, confusion matrix shows that the TP values in the main diagonal box have predominantly light colors. This shows that the model can classify almost all classes that match the correct data in the test data. In the confusion matrix it can also be seen that there are 10 classes that have FN and FP values. Class 19 has 2 FN and Class 13 has 2 FP. Even so, the double FN and FP values in these 2 classes are low. So the model can still classify these 2 classes. The focus of attention is on class 5 and class 20. Because class 5 and 20 have high FN values compared to other classes.

When implementing model 1 in the previous stage, there were 2 letters that were difficult to classify. Namely the letter F which is sometimes classified as the letter B and the letter U which is sometimes classified as the letter R. Based on the confusion matrix in Figure 6, the cause of the difficulty in classifying 2 letters can be identified. It turns out, class 5 as the letter F has a TP with a dark color and a low score, namely 12 out of 95 test data. This is because Class 5 also has an FN value with a dark color and a high value for class 1 as the letter B, which is 83. With TP in class 5 and FN in class 5 against class 1, it shows that the model classifies class 1 as class 5 as many as 83. This also shows that the model is only able to classify class 5 as many as 12 out of 95 test data.

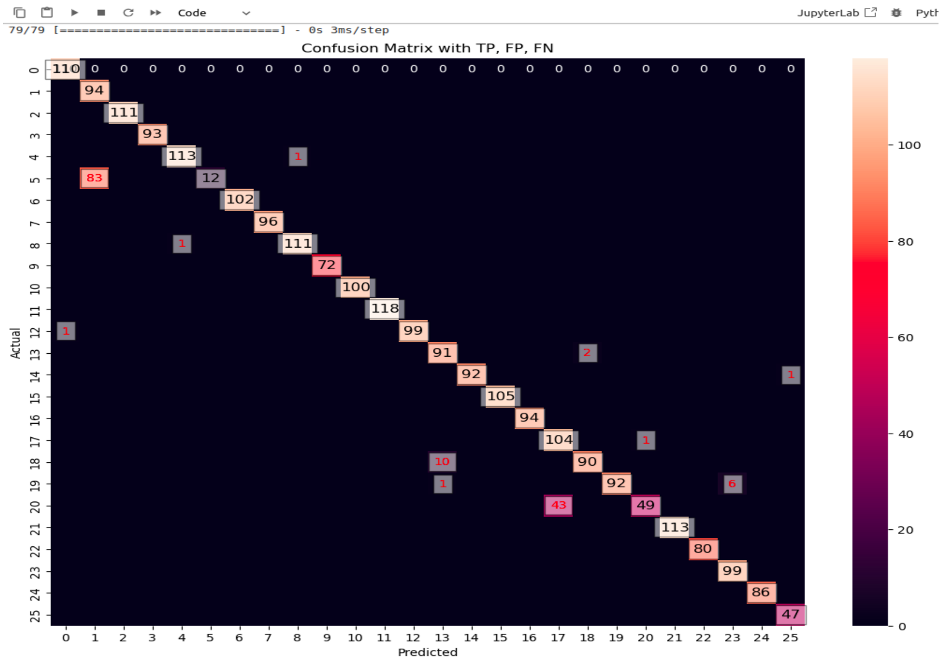

```
79/79 [==============================] - 0s 3ms/step
```

Fig. 6. Model 1 confusion matrix result.

Likewise with class 20 as the letter U. It turns out that based on the confusion matrix, it was found that the model was only able to classify class 20 for 49 out of 92 test data. This is because class 20 as the letter U has an FN for class 17 as the letter R of 43. This means that the model classifies class 20 as class 17 with a total of 43. Complete numerical results from Precision, Recall, F1-Score, Test data, TP, FN, and FP can be seen in Table 1 below.

Based on Table 1 below, it can be concluded that the model 1 classification for each class is almost perfect. This can be seen in the F1-Score and recall values, the majority of which are high. However, based on Table 1, it is found that the FN value of class 5 is 83. This makes the recall of class 5 low, namely 0.13. Meanwhile, the FP value from class 1 is also 83. This makes the precision value from class 1 low, namely 0.53. Because the recall value in class 5 is low, this makes the model imperfect in classifying class 5 as letter F.

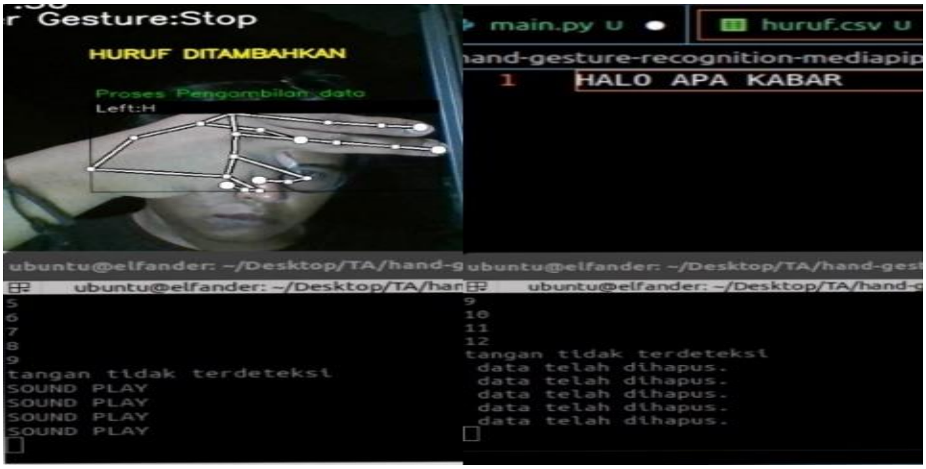
Likewise with class 20. Because class 20 has low recall, the model is not perfect in classifying class 20 as the letter U. The FN value of class 20 as the letter U is the same as the FP value of class 17 as the letter R, namely 43. This makes class 20 sometimes classified as class 17.

Table 1. Model 1 performance results based on Scikit-learn confusion matrix.

Class	Label	Precision	Recall	F1-Score	Test Data	True Positive	False Positive	False Negative
0	A	0.99	1.00	1.00	110	110	1	0
1	B	0.53	1.00	0.69	94	94	83	0
2	C	1.00	1.00	1.00	111	111	0	0
3	D	1.00	1.00	1.00	93	93	0	0
4	E	0.99	0.99	0.99	114	113	1	1
5	F	1.00	0.13	0.22	95	12	0	83
6	G	1.00	1.00	1.00	102	102	0	0
7	H	1.00	1.00	1.00	96	96	0	0
8	I	0.99	0.99	0.99	112	111	1	1
9	J	1.00	1.00	1.00	72	72	0	0
10	K	1.00	1.00	1.00	100	100	0	0
11	L	1.00	1.00	1.00	118	118	0	0
12	M	1.00	0.99	0.99	100	99	0	1
13	N	0.89	0.98	0.93	93	91	11	2
14	O	1.00	0.99	0.99	93	92	0	1
15	P	1.00	1.00	1.00	105	105	0	0
16	Q	1.00	1.00	1.00	94	94	0	0
17	R	0.71	0.99	0.83	105	104	43	1
18	S	0.98	0.90	0.94	100	90	2	10
19	T	1.00	0.93	0.96	99	92	0	7
20	U	0.98	0.53	0.69	92	49	1	43
21	V	1.00	1.00	1.00	113	113	0	0
22	W	1.00	1.00	1.00	80	80	0	0
23	X	0.94	1.00	0.97	99	99	6	0
24	Y	1.00	1.00	1.00	86	86	0	0
25	Z	0.98	1.00	0.99	47	47	1	0

3.3 System Implementation (Last Stage)

The system implementation is carried out by trying to detect 1 sentence with 3 words (HALO APA KABAR). Based on the results of system implementation, the system can classify 1 sentence. To get letters, the TAB keyboard key is used. When the system cannot read finger gestures, the sign "Tangan Tidak Terdeteksi" will appear. When the system saves letters in CSV, the sign "Huruf Ditambahkan" will appear. Huruf Ditambahkan in English are letters added. To produce sound, the enter key is used. The voice produced perfectly matches the Indonesian accent. When the system makes sound, the words "Sound Play" will appear on the system. This implementation process can be seen in Figure 7.



```
1    HALO APA KABAR
```

```
ubuntu@elfander: ~/Desktop/TA/hand-g
ubuntu@elfander: ~/Desktop/TA/han
5
6
7
8
9
tangan tidak terdeteksi
SOUND PLAY
SOUND PLAY
SOUND PLAY
SOUND PLAY
```

```
ubuntu@elfander: ~/Desktop/TA/hand-gest
ubuntu@elfander: ~/Desktop/TA/hand-g
9
10
11
12
tangan tidak terdeteksi
 data telah dihapus.
 data telah dihapus.
 data telah dihapus.
 data telah dihapus.
 data telah dihapus.
```

Fig. 7. System implementation result.

4 Conclussion and Future Scope

This research refers to the use of tensorflow in various applications. In this study, the system was successfully built by combining tensorflow and mediapipe as 2 methods into 1. Mediapipe is used to detect sign language finger patterns and Tensorflow is used to find finger landmark patterns. The results and testing of the study were carried out in 3 stages. In stage 1, it was found that both models could be implemented well. Model 1 has an accuracy of 93% and model 2 has an accuracy of 95%. In model 1, it was found that the letters F and U were difficult to classify. While in model 2, the implementation of the model was perfect. This is because the system can perfectly create circles that form the letters J and Z.

Because model 2 can be implemented well, the focus of the research analysis is only on model 1. In stage 2, it was found that the cause of the 2 letters being difficult to detect in model 1 was because they had high FN values, namely 83 and 43. This makes the recall value of the 2 classes also low, namely 0.13 and 0.53. In the final stage, the system model can be implemented well. This is because the system can detect fingers, classify sign language, and produce sounds using the Text to Voice method. The model in this study was deployed to Tensorflow-Lite. So, in further research, it is expected that the implementation of the model can be done using android.

References

1. W. A. R. Wan Ali, A. Abdul Kassim, and S. M. Mohd Rashid, 'Assistive Technology for Persons with Physical Disabilities: A Literature Review', *International Journal of Academic Research in Business and Social Sciences*, vol. 14, no. 2, Feb. 2024, doi: 10.6007/ijarbss/v14-i2/20823.
2. R. Sandhiya, & V. Saranya. (2021). An application for mute and hearing impaired person to attend phone call. Scientific Hub of Applied Research in Engineering & Information Technology, 1(2), 19–22. https://doi.org/10.53659/shareit/2021/11
3. J. D. Bonvillian, K. Lee, T. T. Dooley, and F. T. Loncke, *Simplified Signs A Manual Sign-Communication System for Special Populations Volume 1: Principles, Background, and Application*, vol. 1. Open Book Publishers, 2020. doi: 10.11647/OBP.0205.
4. R. S. Fauzi, B. Irmawati, and N. Agitha, 'KADARING SIBI (Indonesian Sign System Online Dictionary): Web-based Indonesian Sign System Learning App', in *Proceedings of the First Mandalika International Multi-Conference on Science and Engineering 2022, MIMSE 2022 (Informatics and Computer Science)*, Atlantis Press International BV, 2022, pp. 427–436. doi: 10.2991/978-94-6463-084-8_35.
5. K. Snoddon and M. De Meulder, 'Introduction: Ideologies in sign language vitality and revitalization', *Lang Commun*, vol. 74, pp. 154–163, Sep. 2020, doi: 10.1016/j.langcom.2020.06.008.
6. U. Fadlilah, A. K. Mahamad, a nd B. Handaga, 'The Development of Android for Indonesian Sign Language Using Tensorflow Lite and CNN: An Initial Study', in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Apr. 2021. doi: 10.1088/1742-6596/1858/1/012085.
7. Di. Someshwar, D. Bhanushali, V. Chaudhari, and S. Nadkarni, 'Implementation of Virtual Assistant with Sign Language using Deep Learning and TensorFlow', in *Proceedings of the 2nd International Conference on Inventive Research in Computing Applications, ICIRCA 2020*, Institute of Electrical and Electronics Engineers Inc., Jul. 2020, pp. 595–600. doi: 10.1109/ICIRCA48905.2020.9183179.
8. N. S. Kurian et al., "Transformative Perspectives: AI-Enabled Dubbing Software for Multilingual Content Localization," African Journal of Biological Sciences (South Africa), vol. 6, no. 10, pp. 924–929, 2024, doi: 10.33472/AFJBS.6.10.2024.924-929.
9. D. Ghosh, S. Ghatak, and H. Paul, 'A Proposed Cognitive Framework Model for a Student Support Voice based Chatbot using XAI', 2023, doi: 10.21203/rs.3.rs-2888180/v1.
10. S. Suherman, A. Suhendra, and E. Ernastuti, 'Method Development Through Landmark Point Extraction for Gesture Classification with Computer Vision and MediaPipe', *TEM Journal*, vol. 12, no. 3, pp. 1677–1686, Aug. 2023, doi: 10.18421/TEM123-49.
11. REVA University School of Computer Science and Engineering and Institute of Electrical and Electronics Engineers, *2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT): 5-7 Jan. 2023.*
12. H. Kolivand, S. Joudaki, M. S. Sunar, and D. Tully, 'A new framework for sign language alphabet hand posture recognition using geometrical features through artificial neural network (part 1)', *Neural Comput Appl*, vol. 33, no. 10, pp. 4945–4963, May 2021, doi: 10.1007/s00521-020-05279-7.
13. J. Rakesh, 'COVID-19 and Other Pneumonia Diagnosis Using CNN', *Int J Res Appl Sci Eng Technol*, vol. 10, no. 10, pp. 1519–1525, Oct. 2022, doi: 10.22214/ijraset.2022.47249.
14. B. J. Jo, S. K. Kim, and S. K. Kim, 'Enhancing Virtual and Augmented Reality Interactions with a MediaPipe-Based Hand Gesture Recognition User Interface', *Ingenierie des Systemes d'Information*, vol. 28, no. 3, pp. 633–638, Jun. 2023, doi: 10.18280/isi.280311.

15. S. Shriram, B. Nagaraj, J. Jaya, S. Shankar, and P. Ajay, ‘Deep Learning-Based Real-Time AI Virtual Mouse System Using Computer Vision to Avoid COVID-19 Spread’, *Journal of Healthcare Engineering*, vol. 2021. Hindawi Limited, 2021. doi: 10.1155/2021/8133076.
16. A. Tuan Hoang *et al.*, ‘A review on application of artificial neural network (ANN) for performance and emission characteristics of diesel engine fueled with biodiesel-based fuels’, *Sustainable Energy Technologies and Assessments*, vol. 47, Oct. 2021, doi: 10.1016/j.seta.2021.101416.
17. J. Blechschmidt and O. G. Ernst, ‘Three ways to solve partial differential equations with neural networks — A review’, *GAMM Mitteilungen*, vol. 44, no. 2, Jun. 2021, doi: 10.1002/gamm.202100006.
18. A. Golzari Oskouei, M. A. Balafar, and C. Motamed, ‘EDCWRN: efficient deep clustering with the weight of representations and the help of neighbors’, *Applied Intelligence*, vol. 53, no. 5, pp. 5845–5867, Mar. 2023, doi: 10.1007/s10489-022-03895-5.
19. M. A. Mercioni and S. Holban, ‘The Most Used Activation Functions: Classic Versus Current’, in *2020 15th International Conference on Development and Application Systems, DAS 2020 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., May 2020, pp. 141–145. doi: 10.1109/DAS49615.2020.9108942
20. S. Maharjan, A. Alsadoon, P. W. C. Prasad, T. Al-Dalain, and O. H. Alsadoon, ‘A novel enhanced softmax loss function for brain tumour detection using deep learning’, *J Neurosci Methods*, vol. 330, Jan. 2020, doi: 10.1016/j.jneumeth.2019.108520.
21. D. Ali, M. M. S. Missen, and M. Husnain, ‘Multiclass Event Classification from Text’, *Sci Program*, vol. 2021, 2021, doi: 10.1155/2021/6660651.
22. M. Heydarian, T. E. Doyle, and R. Samavi, ‘MLCM: Multi-Label Confusion Matrix’, *IEEE Access*, vol. 10, pp. 19083–19095, 2022, doi: 10.1109/ACCESS.2022.3151048.
23. A. Tharwat, ‘Classification assessment methods’, *Applied Computing and Informatics*, vol. 17, no. 1, pp. 168–192, 2018, doi: 10.1016/j.aci.2018.08.003.
24. K. J. Tsai *et al.*, ‘A High-Performance Deep Neural Network Model for BI-RADS Classification of Screening Mammography’, *Sensors*, vol. 22, no. 3, Feb. 2022, doi: 10.3390/s22031160.
25. A. A. Aldino and H. Sulistiani, “Decision Tree C4.5 Algorithm for Tuition Aid Grant Program Classification (Case Study: Department of Information System, Universitas Teknokrat Indonesia),” 2020. doi: 10.21107/edutic.v7i1.8849