P-ISSN 2586-9000
E-ISSN 2586-9027
Homepage : https://tci-thaijo.org/index.php/SciTechAsia
Science & Technology Asia
Vol.29 No.3 July - September 2024
Page: [94-114]
Original research article

Real-Time Classification Improvement of Indonesian Sign System Letters (SIBI) Using K-Nearest Neighbor Algorithm

Oktaf Agni Dhewa[1,*], Safitri Yuliana Utama[1], Aris Nasuha[1], Teddy Surya Gunawan[2], Gilang Nugraha Putu Pratama[3]

[1]*Department of Electrical and Electronics Engineering, Vocational Faculty, Universitas Negeri Yogyakarta, Yogyakarta 55651, Indonesia*
[2]*Department of Electrical and Computer Engineering Department, Kulliyyah of Engineering, International Islamic University Malaysia, Gombak Selangor 53100, Malaysia*
[3]*SAM Laboratory, Wonosobo 56312, Indonesia*

Received 8 November 2023; Received in revised form 27 June 2024
Accepted 9 July 2024; Available online 25 September 2024

ABSTRACT

Indonesian Sign Language (SIBI) is a vital means of communication for individuals with hearing impairments. The automatic translation from spoken language to SIBI presents challenges in accurately predicting sign characters. The information transfer process becomes biased when system predictions are incorrect. The current approach lacks accuracy due to data variations that may lead to character similarities. This research addresses this issue with an improved method incorporating linguistic features and contextual information. A novel approach is introduced to enhance SIBI character predictions using the K-Nearest Neighbor (K-NN) algorithm. The K-NN algorithm is employed to predict the most suitable SIBI character based on the similarity of linguistic features between input speech and existing data. This research compares distance metrics such as Euclidean, Manhattan, and Chebyshev to determine the optimal number of nearest neighbors (K) for achieving the most accurate outcomes. Experimental results employing 200 data points per label yielded satisfactory average predictions for each label. The experiments underscore the effectiveness of the K-NN model utilizing the Chebyshev distance metric with K=7 on the 200 data labels, as it provided excellent probability results for each label.

Keywords: Distance Metric; Lazy learner system; Machine learning; Sign language

***Corresponding author:** oktafagnidhewa@uny.ac.id
doi: 10.14456/scitechasia.2024.49

1. Introduction

Hand sign language is one form of communication used by people with disabilities, including the deaf, speech impaired, blind, and even general people [1]. The use of hand sign language certainly has variations and different patterns of views in Indonesia, which has more than 1,300 ethnic groups recorded by the Statistics Indonesia (Badan Pusat Statistik) in 2010 and has various regional languages [2]. Thus, using sign language has different meanings in each usage pattern. However, even though diversity provides a difference, Indonesia has a national language, namely Indonesian, so it can make it easier for someone to communicate with people from different cultures. In 1981, several deaf and blind activists formed the Indonesian Disabled Lovers Movement (Gerkatin) in Surabaya [3], which later developed the Indonesian Sign Language System, known as Sistem Isyarat Bahasa Indonesia (SIBI) in Bahasa Indonesia, which was adapted from ASL American Sign Language [4] which the Indonesian government then inaugurated in dated 19 August 2005, use of Indonesian Sign Language. Both SIBI and ASL encompass hand signs that correspond to each letter of the alphabet, allowing for fingerspelling. Implementing SIBI provides support and convenience for people with disabilities and ordinary people as a form of communication. However, despite this, sign language still faces many challenges in its implementation, some of which are the lack of public understanding and awareness of the rights of deaf and blind people and, on the other hand, the lack of accessibility of information so that not many people understand the use of SIBI [5, 6]. Some public facilities already have graphics and are even specifically for people with disabilities [6], so research was carried out to produce a form of application that is easily accessible for individuals with smartphones. This technology can detect sign language with the help of artificial intelligence, built by several studies. However, some of them still have low accuracy and high prediction errors because there are letter signals that have very similar shapes. This research is similar to discussing the detection of Indonesian Sign Language (BISINDO) [7]. Problems in false detection often occur. Some things are caused by poor data pre-processing, so the model has incorrect predictions in its learning. In addition, similarities between letters are a challenge in terms of detection [8]. Therefore, efforts are still needed to detect the Indonesian language sign system more accurately. Henceforth, in this research, we propose a solution using the K-Nearest Neighbor (K-NN) machine learning algorithm to tackle these problems. Despite the fact that K-NN has several limitations, such as sensitivity to outliers and poor performance on large datasets [9, 10], it still stands out because it is simple and easy to understand. This algorithm does not require learning, can adapt to changes in data, can handle imbalanced data, is flexible in choosing the distance matrix, and performs well on small datasets. This makes it possible to implement real-time classification for SIBI compared to previous methods.

2. Related Works

Sign Language is an important aspect for people with hearing and speech impairments. This language makes their communication easier and more complete. Therefore, sign language also needs to be learned easily by people who do not have disabilities in interacting with deaf and speech impaired people. The learning process, which

is quite difficult for people without disabilities, has encouraged the development of learning using a technological innovation approach to make it easier. One of these developments is the use of artificial intelligence technology applied to computer vision computing to detect sign language signs using hands.

Designing and implementing this technology is certainly not an easy thing. Various researchers have carried out their findings in stages to achieve perfection, as done by Ridwan G et al. (2019). Their research applies the Naive Bayes classification algorithm to Leap Motion to detect Indonesian language signs (SIBI). This algorithm works by estimating probability values for each class of data given, then predicting a class based on these probability values. The evaluation results showed an average accuracy of 96%, but several prediction errors occurred in the letters ”M”, ”N”, and ”O”, as well as the letters ”Z”, ”U”, and ”L”. Prediction errors also occurred on the letters ”J” and ”L”, caused by input data from Leap Motion not being able to recognize hidden fingers [9].

Sign language creation methods continue to develop in the world of artificial intelligence. This is proven by research conducted by Afifah, et al. (2021). This research detects letter features in the Indonesian Sign Language System (SIBI) using the chain method. The letter detection process involves pre-processing, edge detection, image extraction, and letter matching. In the segmentation process, the Manhattan distance method is used, followed by converting the RGB image to grayscale. Grayscale values are mapped to the image shape which produces values 1 and 0 to obtain a probability level. The training data evaluation results show an average accuracy of 91% with input in the form of images [10].

In the following years, sign language detection continued to develop. Putri, et al. (2022) conducted a real-time Indonesian sign language detection study using long short-term memory (LSTM). The language model used is Indonesian Sign Language (BISINDO). The collected data is classified by LSTM and detection of movement frames starting from the hands, face and body using Media Pipe Holistic. This research uses 30 BISINDO sign vocabularies, and the data is labeled with an array which is then divided into 95% training data and 5% testing data. The research results show that testing the first 10 classes using bidirectional LSTM with 1000 epochs, 64 hidden layers, and batch size 32 produces an accuracy of 92%. However, testing 30 classes with 2 LSTM layers trained for 500 epochs, 64 hidden layers, and batch size 64 produces an accuracy of 65%, which shows a decrease in quality [11].

Another research using machine learning methods in the introduction of Indonesian Sign Language (SIBI) was carried out by Imam, et al. (2022). The models used are Random Forest and Multinomial Logistic Regression. This gesture recognition detection includes 24 letters, without the letters ”J” and ”Z” because the hand signals for these two letters are dynamic. The Random Forest model has an average accuracy of 97.19% with a number of decision trees (trees) of 750. However, there are several errors in the letters ”K” and ”R” which have similar hand signals. In addition, the Multinomial Logistic Regression model produces an average accuracy of 95.73%, with some errors occurring when predicting the letters ”U” and ”V” [12].

Several artificial intelligence development models that have been carried out

still have shortcomings, such as computing which takes quite a long time when running in real-time and some letters of the alphabet that cannot be detected properly, so that the learning process is less than optimal. Based on this, research related to sign language sign detection continues with the development of more complex but lightweight methods to obtain the best model. One method that has this potential is K-Nearest Neighbour (K-NN), as researched by Rivan, et al. (2020).

In their research, the process of detecting language signs from American Sign Language (ASL) for the letters A to Y, without the letters J and Z, using the K-Nearest Neighbour (K-NN) classification method showed the highest accuracy of around 72%. This K-NN model mechanism utilizes the Histogram of Oriented Gradients (HOG) to calculate gradient values in certain areas of an image. Then, the HOG results are linearized using Linear Discriminant Analysis (LDA) to obtain a matrix. Test results using Euclidean, Manhattan, and Chebyshev distance matrices with k values of 3, 5, 7, 9, and 11 for each distance show the highest precision, recall, and accuracy values at k=3, k=5, and k =11 [13].

Similar research was also conducted by Hasma, et al (2022). This research focuses on recognizing Indonesian gestures using the SURF and K-Nearest Neighbour (K-NN) algorithms. The detection carried out includes Indonesian sign system alphabets (SIBI) and numbers. The test results with test data show that the value K=7 obtains the highest accuracy of 90%, the value K=8 obtains the highest accuracy of 88%, the value K=9 obtains the highest accuracy of 86%, and the value K=10 obtains the highest accuracy of 87 .5% [14].

Using the K-NN method to recognize and detect language signs is the right step. A language sign detection system can be carried out accurately and efficiently in real-time in recognizing various language signs in the world. The results of this research were able to classify and recognize different hand shapes and colors well. In addition, the K-NN method for recognizing language signs has a model with a high level of precision and can be developed further [15, 16].

Therefore, this research is focused on developing a model for recognizing Indonesian language signs (SIBI) with efforts to improve the prediction of each letter using the Supervised Learning method, namely the K-Nearest Neighbour (K-NN) algorithm. This research compares models with Euclidean, Manhattan, and Chebyshev distance metrics to obtain the highest accuracy. Data collection was carried out based on pixel coordinates of 5 important points on the hand (landmarks) and utilized the Euclidean formula for recognition of linear line patterns.

3. SIBI Classification Attributes System

Improved SIBI detection classification is realized through a portable physical system. The system consists of several integrated component attributes, including electronic components that are executed with programming instructions and displayed in a GUI application that is packaged with a portable mechanical design that is easy to operate.

3.1 Electronics design

The electronics are designed to place the position where each component needs to be installed. The components required are HDMI LCD (Liquid Crystal Display), Raspberry Pi, Keyboard, Mouse, and Camera, whose design is shown in Fig. 1.

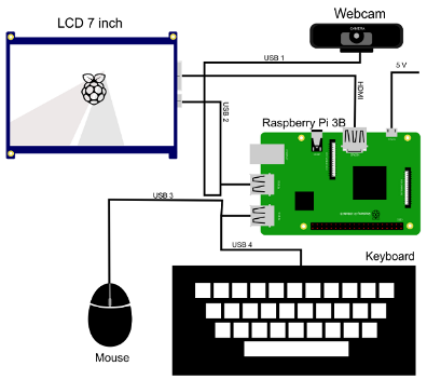

Fig. 1. SIBI electronics design.

The Raspberry Pi 3B is equipped with an ARM Cortex-A53 Quad Core processor with a speed of 1.2 GHz 64-bit, providing adequate performance to run applications and light to medium-level computing tasks. The Raspberry Pi 3B has the lowest standard in use. It has 1 G.G.B.B. of RAM and has 4 USB ports, which are connected to the camera, the second port connected to the keyboard, the third port is connected to the mouse, and finally, the last port is connected to the voltage input from the LCD and the HDMI port. The Raspberry Pi 3B stores algorithm data and model results in this project. With the help of the detection Python library, the detection program can be run. Meanwhile, the HDMI LCD (Liquid Crystal Display) port is the monitor used to display the display on the Raspberry Pi in this project. This monitor is 7 inches or around 164.9 mm × 124.2 mm and is capacitive [17].

3.2 Software and logical design

Software is a collection of instructions that has a role as a system designer in building alphabet detection in this final project. This is necessary because it requires components from the Python language, which concerns the use of libraries. The libraries process data, build machine learning models, and create graphical user interfaces (GUI).

3.2.1 Library package

The library used in this project is one of the requirements in developing the Indonesian Sign System (SIBI) alphabet letter detection model as well as creating a Graphical User Interface (GUI) display, including,

- Python 3.8.12 is one of the choices for this project because it is one of the highest versions of Python 3.8 provided by the Raspberry pi model 3B. It is also available for use with the mediapipe library, which Google developed.
- MediaPipe is an open-source library developed by Google, and one of the features of this library is hand detection. The library is data that utilizes the ID value of certain points on the hand.
- Numpy, or what is usually called Numerical Python, is used to convert Numpy array values to strings in real-time model testing to display probability values.
- OpenCV is used in camera usage; besides, it is used to display the hand id value from MediaPipe.
- Pillow replaces the PIL (Python Imaging Library) library, which calls signal images that will be displayed in the application.
- Scikit Learn is a library that has several machine-learning algorithms. This library requires the K-Nearest Neighbors algorithm used in this project to produce detection.

- Pandas is a Python library used for data analysis and manipulation. In this project, it is used to display processed data to show changes to previous data.
- Comma Separated Values (CSV) is one of the libraries in Python. This library is used to store data in table form. In this project, this library retrieves numerical data, which is automatically saved in a file in .csv format.
- Math is a Python library used in the process of mathematical operations. In this project, it is used to calculate the distance between finger IDs using several mathematical formulas.
- Tkinter is an option for creating Graphical User Interfaces (GUI). This library is one of the standard Python libraries that can provide a simple appearance for creating applications and can function.

3.2.2 GUI design

This application design is one of the supports in testing the model used in this project. In making the application, the Tkinter library from Python is needed to create several layouts that will display images and cameras. The application layout design is shown in Fig. 2.

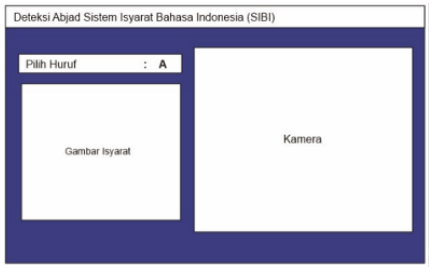

Fig. 2. SIBI graphical user interface.

The GUI has layout provisions with a detailed application size of 685 × 345 in pixel units, a camera size of 250 × 300 in pixel units, a signal image size of 200 × 198 in pixel units, and the size of the dropdown list adjust the size of the design according to the length and width of the camera. The size is influenced by data collection using a camera with that resolution. In addition, other layouts adapt to the size of the window.

3.3 Mechanical design

The mechanical design designs how the layout of the electronic components will be installed using the box as a container. The following is a design plan for the box shown in Fig. 3.

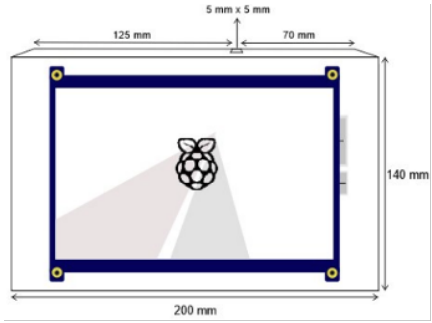

Fig. 3. Top view mechanical design.

The front view design shows the placement of the HDMI LCD, where the top shows a 5 mm × 5 mm hole used for the camera cable route, then the side view design is shown in Fig. 4.

In the edge area, a hole measuring 25 mm x 5 mm is used to route the HDMI LCD cable, mouse, keyboard, and power supply.

4. K-Nearest Neighbor Model

K-NN demonstrates strong consistency properties. As the dataset size becomes very large, the error rate of the two-class K-NN algorithm is assured to be no more than double the Bayes error rate,

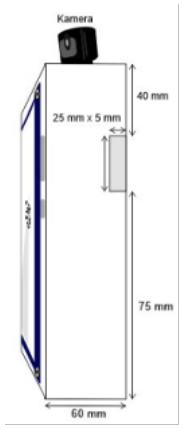

Fig. 4. Side view mechanical design.

which is the lowest possible error rate based on the data distribution. Cover and Hart [18] establish an upper bound error rate for multi-class k-NN classification, such as

$$R^{\star} \leq R_{KNN} \leq \left(2 - \frac{MR^{\star}}{M-1}\right), \qquad (4.1)$$

where $R^{\star}$ represents the Bayes error rate, which is the minimum possible error rate, R_{KNN} denotes the asymptotic K-NN error rate, and M is the number of classes in the problem. This bound is considered tight because both the lower and upper limits can be achieved by some distribution [19]. When $M = 2$ and the Bayes error rate $R^{\star}$ approaches zero, this limit simplifies to no more than twice the Bayes error rate.

The upper bound error rate established by Cover and Hart provides predictive reliability and benchmarks for K-NN's worst-case scenario. It aids in algorithm selection and guides practitioners on when K-NN will perform well. Understanding these bounds allows for targeted improvements and optimizations, enhancing the performance of K-NN. Additionally, it offers a solid theoretical foundation, instilling confidence in K-NN's robustness and consistency across various classification tasks.

The data used in this research is numerical data for detecting signed alphabet letters using the Indonesian Sign Language System (SIBI) guidelines [20]. Numerical data is obtained from reading distance IDs on fingers. With the help of the MediaPipe library from Google, hands can be detected and display 21 hand ID features. Another reason for using MediaPipe is that hand detection by MediaPipe is the same for the right hand and left hand, so there is no need to take 2 data with the right hand and the left hand. The following hand ID displayed by one of the features of MediaPipe is shown in Fig. 5.

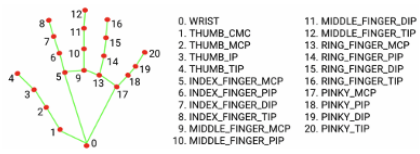

Fig. 5. MediaPipe of hand index.

ID[4] is Thumb-Ip, ID[8] is Index-Finger-Tip, ID[12] is Middle-Finger-Tip, ID[16] is Ring-Finger-Tip, and ID[21] is Pinky-Tip.

The index is sufficient to represent the variable value for each alphabet class. Data retrieval begins with calculations between the Euclidean formula's IDs[22].

$$\text{Euclidean distance} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}. \qquad (4.2)$$

This formula will form a linear line aimed at pattern recognition and collect data using a camera window with a ratio of 250 × 300-pixel units [23]. So, the data value of each variable is very dependent on the camera's coordinate point ID. Variable values will be stored in the .csv file, following an example of retrieving numerical data shown in Fig. 6.

In the camera window, all IDs are displayed, but the coordinate calculation only occurs for IDs 4, 8, 12, 16, and 20. The

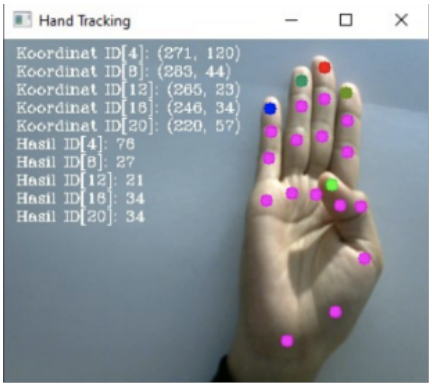

Fig. 6. Data labeling.

calculation results will be converted into integer values saved in the .csv file, where retrieval will occur. The data processing is sequentially shown in the following process flow,

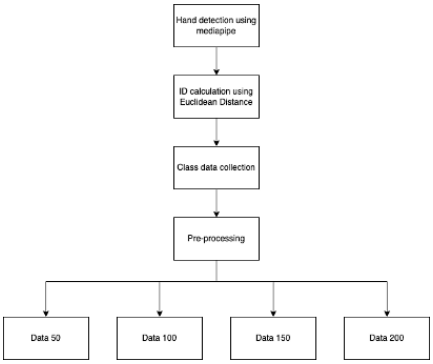

Fig. 7. Data processing.

4.1 Data modelling

Creating a detection model requires an algorithm for which the author chose to use K-Nearest Neighbors (K-NN). The K-NN algorithm is a supervised learning algorithm that requires data with labels [8]. The choice of this algorithm is due to adjusting data with distance calculation properties. In this research, a comparison of metric distance calculations from introducing new data patterns was carried out, namely using Euclidean, Chebyshev, and Manhattan. Comparison of Distance metric search methods aims to obtain the best standard feature parameters [24].

Apart from the factor of using distance metrics, something that needs to be considered is the large influence of the K value. The K value is obtained using the Rule of Thumb technique, where the results of this technique depend on the number of classes. This technique considers the number of 26 classes by rooting them to produce a value of 5.09. Selecting numbers if the class value is even, then the K value is odd so that there are no misunderstandings in calculating data using distance [25]. The sequential creation of model data is shown in the flow diagram in Fig. 8.

K-NN modeling data starts from data pre-processing. This stage aims to clean the data from outlier values and can also be called data normalization using the Z-Score method. Then, the next stage of data training uses the K-Nearest Neighbors (K-NN) algorithm, where the data training process is carried out for each desired K value and the distance metric that will be used. Then, the training results produce a model in the .joblib format, which is better than the .pkl model or what is often called pickle because the .joblib model is more efficient in-memory processing and faster in the model load process. The detection model results will be evaluated, and if the model evaluation results are not as expected, the model will be trained with different K values and distance metrics.

5. Unit Testing Scenario

System testing is carried out by processing the data acquisition results by looking at the part of the value that is out of scale. The scale value uses the Z-Score method, where the scale limit value is selected at a maximum of 3 and a minimum

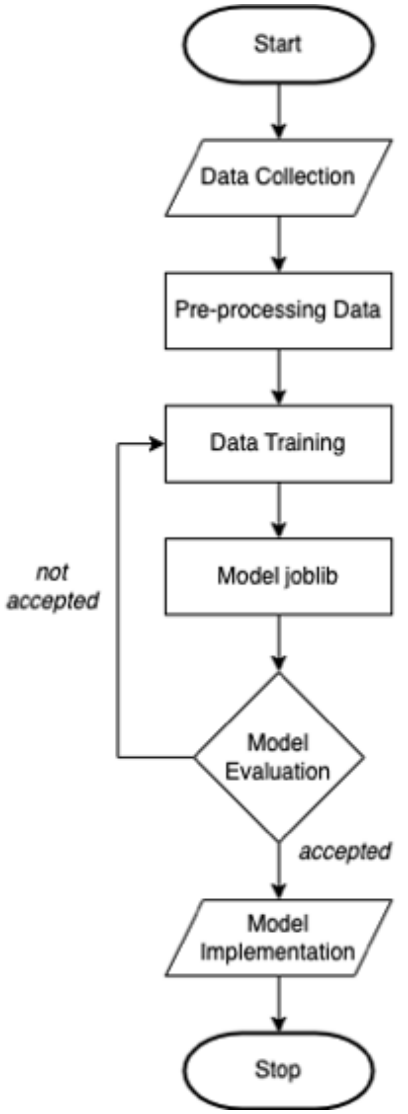

Fig. 8. K-NN data model.

of -3. The value will be displayed in graphical form for each label, where the graph will display outlier values on the Z-score scale. This process is called pre-processing, which means the data is processed to be cleaned, and the cleaned data can be used in creating models [26]. Pre-processing is carried out with the help of Microsoft Excel, where the Z-score method is formulated as follows:

$$Z = \frac{x - \mu}{\sigma}, \qquad (5.1)$$

where x is the variable value, μ is the average of all data in one variable, and σ is the standard deviation. The process of testing the collected data is shown in Fig. 9

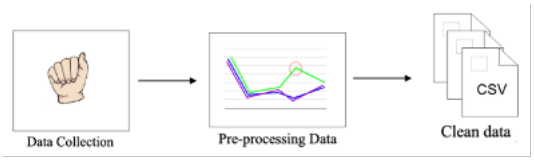

Fig. 9. Data testing schema.

The model results will be tested with test data to produce model accuracy values. Apart from that, real-time testing was also carried out using a camera to detect the probability results for each label directly.

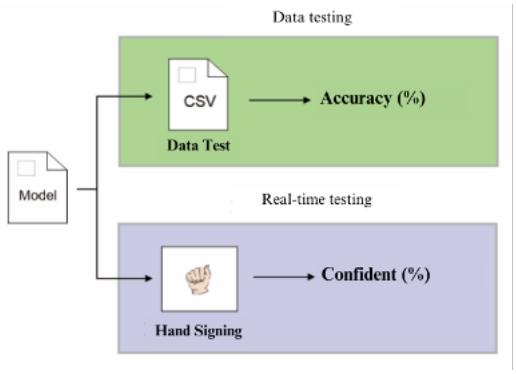

Fig. 10. Model testing schema.

The model testing design was carried out on ten people, of which five people were male and five were female.

6. Result and Discussion

6.1 Data generating

The data collected is a collection of numbers calculated using the Euclidean formula, where numbers are obtained from the coordinate distance of one index to another. The use of indexes in this research takes five indexes on hand using the MediaPipe library. The hand indices used are 4, 8, 12, 16, 20. Calculating the index distances

produces four distance features so that data from the four distances is mapped to a camera resolution of 250 pixels x 300 pixels.

The resulting data will be saved in a CSV format file with approximately 200 labels per label. One of the results of the labeling feature is shown in Table 1.

Table 1. The 4-feature dataset model.

Distance 1	Distance 2	Distance 3	Distance 4	Label
75	33	28	31	A
72	33	28	31	A
75	33	28	31	A
75	23	22	20	A
64	26	24	25	A
65	28	25	25	A

In addition, we also generated labeling data for the 5 dataset features. The fifth feature is duplicated from the last distance of the 4 dataset features. Therefore, the letter A detection data will have features as in Table 2.

Table 2. The 5-feature dataset model.

Distance 1	Distance 2	Distance 3	Distance 4	Label
75	33	28	31	A
72	33	28	31	A
75	33	28	31	A
75	23	22	20	A
64	26	24	25	A
65	28	25	25	A

The data that has been collected will go through a pre-processing process, which at this stage uses the Z-score as a calculation of limiting value scales with a maximum range of 3 and a minimum of -3. Another method is to look at the graph of the relationship between variables for each class. Results that are out of scale or appear as outliers on the graph will be deleted or replaced. The graphic image display is shown in Fig. 11.

After the pre-processing period, data testing will be done by dividing the data

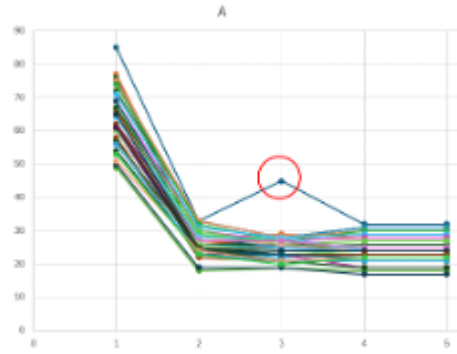

Fig. 11. The outlier value of the third variable of A letter.

concerning data amounts of 50, 100, 150, and 200. This data variation is used to find model results that can detect all the alphabets well.

6.2 SIBI data modelling

This research uses numerical data as training data. Data usage is divided into four types, namely 50 data per label, 100 data per label, 150 data per label, and 200 data per label. This data has gone through a pre-processing stage, which is used to look for outlier values [27]. These values are numbers that deviate from what they should be and are detected by using the graphs' data plot method. Then, trimming the data using the Z-Score method can eliminate unnecessary data in the range of 3 to -3. Then, each piece of data is divided into training data and test data. The proportion of division is 80% training data and 20% training data without using Cross-Validation in the division. The distribution of data for each type of data is explained in Table 3.

Table 3. Data sharing.

Data	Total of Data	Data Learning	Data Test
50	1300	1040	260
100	2600	2080	520
150	3900	3120	780
200	5200	4160	1040

After the training data, we will enter the data training stage. At this stage, K-Nearest Neighbors (K-NN) will not train each data feature but will carry out a lazy learner process or a process where the features in the training data and their labels will be stored in a model with the .joblib format. This format is used because it has a deserialization process where if the model file is used in testing, it will convert the data back into its original form, so it does not require a retraining process and is an alternative library to pickle which has been optimized for storing large data. This stage uses three distance metrics: Euclidean distance, Manhattan distance, and Chebyshev distance. Apart from using these metrics, this process uses the K value, greatly influencing the model performance results. The K value used is obtained from the Rule of Thumb method [10]. The research has 26 classes, which, if using this technique, will produce a value of 5.09, and as a form of comparison, this research uses 3 K values , which take K=3 and K=7. The data process was trained nine times on each type of data. So, using four types of data will produce 36 models.

6.3 Model testing result using data test

The training model will be tested with test data, where the previous data was trained 36 times with different distance parameters and K values. This test produces accuracy, which indicates how well the model performs. The following is the model testing results from a dataset of 4 features and 5 features with test data. The accuracy results of each model are displayed in Tables 4-5.

The accuracy above shows what percentage of the model can predict a label as a whole. The results in the table are influenced by the K value and the use of the distance metric [28]. So, they can be plotted by following the K value, such as Figs. 12-14 for graphic results using 4 dataset features and Figs. 15-17 for test results from using 5 dataset features.

Table 4. The accuracy of the 4 and 5 features dataset model.

Model	K-NN (Accuracy in %)						
	4-features				5-features		
	K	E	M	C	E	M	C
Data 50	3	99	99	99	99	99	99
	5	99	99	99	99	98	98
	7	98	98	98	98	97	97
Data 100	3	99	99	99	99	99	99
	5	98	99	98	99	98	98
	7	98	99	98	98	98	99
Data 150	3	99	99	99	98	99	99
	5	98	98	98	98	98	98
	7	98	98	98	98	98	98
Data 200	3	98	98	98	99	98	98
	5	98	98	98	98	98	98
	7	97	97	97	97	97	98

Notes: E = Euclidean, **M** = Manhattan, **C** = Chebyshev

Table 5. The 50 data per label testing result of the 4-feature dataset.

K	Distance Matrix	Results
3	Euclidean	• The letters A, M, and R are difficult to detect. • The letter H is not detected.
	Manhattan	• The letters C, E, Q, T, and U are difficult to detect.
	Chebyshev	• The letters A and C are difficult to detect.
5	Euclidean	• The letters E and U are difficult to detect.
	Manhattan	• The letter A is difficult to detect.
	Chebyshev	• The letters C, E, and Q are difficult to detect. • The letter A is not detected.
7	Euclidean	• The letters H and G are difficult to detect.
	Manhattan	• The letters C, G, H, L, and U are difficult to detect. • The letters A, E, and Q are not detected.
	Chebyshev	• The letter A is not detected. • The letters C, E, H, and Q are difficult to detect.

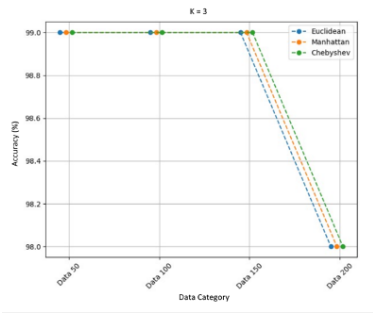

Fig. 12. Graph of 4 feature dataset model with K=3.

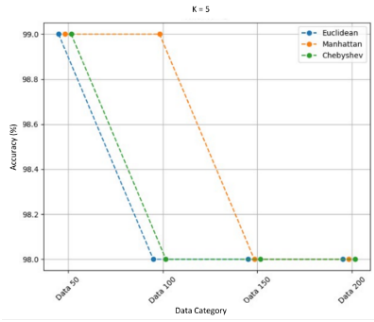

Fig. 13. Graph of 4 feature dataset model with K=5.

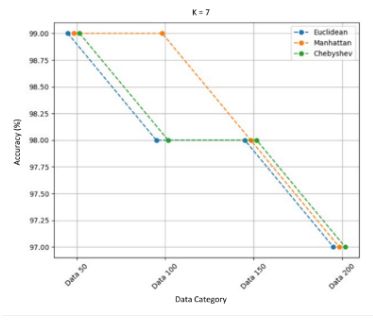

Fig. 14. Graph of 4 feature dataset model with K=7.

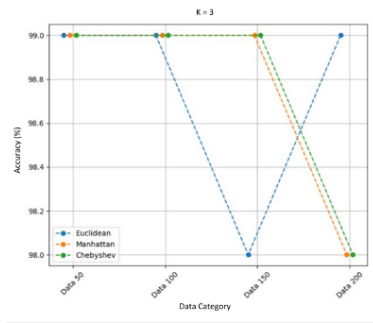

Fig. 15. Graph of 5 feature dataset model with K=3.

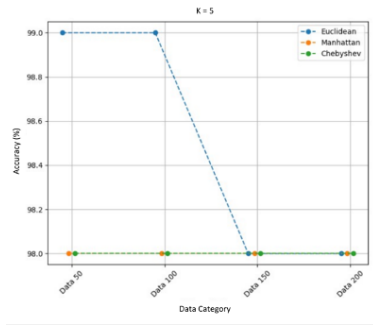

Fig. 16. Graph of 4 feature dataset model with K=5.

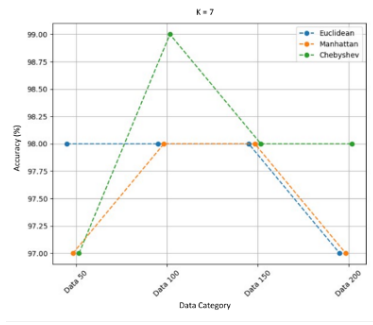

Fig. 17. Graph of 4 feature dataset model with K=7.

6.4 Real-time testing result

The selection of a good model is determined by real-time testing of the models when using a camera. Probability metrics show how likely the hand signal is true, ranging from 0% to 100% [29]. This testing aims to select the best model to be used and applied to the application that has been designed. Real-time testing carried out subjectively by the author shows the probability of each letter shown in Fig. 18.

The test carried out was in the form of the letter A signal, which in the description shows that the probability of the letter

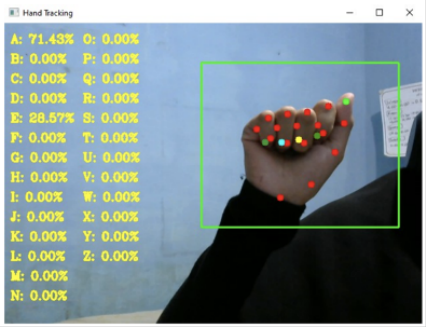

Fig. 18. A letter sign detection probability.

Table 6. The 50 data per label testing result of the 5-feature dataset.

K	Distance Matrix	Results
3	Euclidean	• The letters A, D, K, and E are difficult to detect.
	Manhattan	• The letters H and U are not detected.
	Chebyshev	• The letter H is very difficult to detect. • The letter H is difficult to detect.
5	Euclidean	• The letters A, L, K, and X are very difficult to detect.
	Manhattan	• The letter A is difficult to detect.
	Chebyshev	• The letter H is not detected. • The letters Q and U are difficult to detect.
7	Euclidean	• The letter H is not detected.
	Manhattan	• The letters A, E, K, Q, U, and Z are difficult to detect.
	Chebyshev	• The letters E, H, J, Q, and Z are difficult to detect.

A is 71.43% and the letter E is 28.57% from an accumulation of 100%, which means that two labels predict the hand signals. However, the correct prediction results will be obtained by the label. The one with the highest score is the letter A. Overall, the models tested are detailed in the tables that will be presented. Model testing results are displayed in Table 5 for 50 data per label when using a 4-feature dataset and Table 6 when using a 5-feature dataset.

The 50 data per label and 4 dataset features give poor results in detecting each label, where, on average, the labels can only be detected from a very long distance with

Table 7. The 100 data per label testing result of the 4-feature dataset.

K	Distance Matrix	Results
3	Euclidean	• The letters E, H, and L are difficult to detect. • The letter U is not detected.
	Manhattan	• The letters A, H, Q, and U are difficult to detect. • The letter E is not detected.
	Chebyshev	• The letter M can detect if the hand is brought close to the camera. • The letters C, E, L, Q, and U are difficult to detect.
5	Euclidean	• The letter U is not detected. • The letter R is difficult to detect
	Manhattan	• The letter U is not detected. • The letters E and T are difficult to detect
	Chebyshev	• The letter A is not detected. • The letters C, E, and Q are difficult to detect
7	Euclidean	• The letters H, K, and G are difficult to detect. • The letters H, L, and M can detect if the hand is brought close to the camera.
	Manhattan	• The letter M can detect if the hand is brought close to the camera. • The letter H is difficult to detect
	Chebyshev	• The letters T and X are not detected. • The letter D is difficult to detect. • The letter E can detect if the hand is brought close to the camera.

the camera. Some letters, such as the letter H, cannot be detected, and many letters are still very difficult to detect. With 50 data per label, 5 dataset features, many of which are still difficult to detect. The letters U and H were not detected, and many letters could not be predicted well. The similarity between the two is that the detection results for each label will be difficult to find even though the accuracy of the Data 50 model is very high. This is due to the minimal variation in the dataset. Furthermore, the label prediction results on 100 data per label are displayed in Tables 7-8.

Data from 100 datasets with 4 features shows results that are not much different from data from 50, where some let-

Table 8. The 100 data per label testing result of the 5-feature dataset.

K	Distance Matrix	Results
3	Euclidean	• The letter H often predicts the letter U. • The letter R often predicts the letter U.
	Manhattan	• The letter E often predicts the letter A. • The letter G often predicts the letter U. • The letter Q often predicts the letter X.
	Chebyshev	• The letter R often predicts the letter U. • The letter H often predicts the letters U and R. • The letter R often predicts the letter U.
5	Euclidean	• The letter A often predicts the letter E. • The letter E often predicts the letter A. • The letter U is not detected.
	Manhattan	• The letter D often predicts the letter L. • The letter H often predicts the letter U. • The letter J often predicts the letter I.
	Chebyshev	• The letter R often predicts the letter J. • The letter D often predicts the letter L. • The letter H is difficult to detect. • The letter Q often predicts the letters X and U.
7	Euclidean	• The letter R is not detected.
	Manhattan	• The letter E often predicts the letter A. • The letter G often predicts the letter Q. • The letter L often predicts the letter D. • The letter R is difficult to detect.
	Chebyshev	• The letter A often predicts the letter E. • The letter D often predicts the letter L. • The letters E, G, and Q are difficult to detect

ters are not detected, and some letters are only detected at very close distances and cannot be detected at longer distances. In the 100 data per label dataset, there are 5 features. This data type has improvements in that all labels can be detected but with a low probability value. So, label detection often occurs in prediction errors with letters that have similar shapes. Retesting was carried out on 150 data types, and. The results are listed in Tables 9-10.

Table 9. The 150 data per label testing result of the 4-feature dataset.

K	Distance Matrix	Results
3	Euclidean	• The letters H and M can detect if the hand is brought close to the camera. • The letters D and U are difficult to detect.
	Manhattan	• The letter E is difficult to detect.
	Chebyshev	• The letter Q is difficult to detect. • The letter U is detected out of the ideal distance.
5	Euclidean	• The letters M and Q are not detected. • The letter H is difficult to detect.
	Manhattan	• The letter E is not detected. • The letter R is difficult to detect.
	Chebyshev	• The letters A, E, and Q are difficult to detect. • The letters R and U are detected at an ideal distance.
7	Euclidean	• The letter E is difficult to detect.
	Manhattan	• The letter H can detect if the hand is brought close to the camera. • The letter E is difficult to detect.
	Chebyshev	• The letters D and E are difficult to detect. • The letter U is not detected.

150 data per label dataset with 4-features shows better results than previous data types because only one label from the model is not detected. However, some letters can be detected at close range and cannot be detected from a distance far enough from the camera, and some letters still have difficulty detecting. Then, the results from testing 150 data on a dataset of 5 features where this type of data only has three letters with a low probability value. However, some of these letters are still difficult to detect and tend not to be detected properly. Some suspects refer to the amount of data being insufficient or the distribution of data

Table 10. The 150 data per label testing result of the 5-feature dataset.

K	Distance Matrix	Results
3	Euclidean	• The letter R is not detected. • The letter G is difficult to detect.
	Manhattan	• The letter R often predicts the letter U.
	Chebyshev	• The letter H often predicts the letters U and R.
5	Euclidean	• The letters U and I are difficult to detect.
	Manhattan	• The letter E is difficult to detect.
	Chebyshev	• The letter R is difficult to detect.
7	Euclidean	• The letter R is not detected.
	Manhattan	• The letter J is not detected. • The letter R often predicts the letters U and H.
	Chebyshev	• The letter R is difficult to detect.

values being uneven so that some points are not recognized. Then, testing was carried out on the last type of data, namely 200 data per label, which was tested with the results described in Tables 11-12.

Table 11. The 200 data per label testing result of the 4-feature dataset.

K	Distance Matrix	Results
3	Euclidean	• The letters A, D, E, L, R, and X are difficult to detect.
	Manhattan	• The letters A and R are detected at an ideal distance.
	Chebyshev	• The letters A, D, and R are difficult to detect.
5	Euclidean	• The letter U is not detected.
	Manhattan	• The letters D, E, and Q are difficult to detect.
	Chebyshev	• The letters A and U are not detected. • The letters E, H, and X are difficult to detect.
7	Euclidean	• The letters E and Q are difficult to detect.
	Manhattan	• The letter Q is not detected.
	Chebyshev	• The letters D, E, R, and Q are difficult to detect.

The best model results in predicting are the features in the model with 4 features of the Manhattan dataset with K=3 which includes 200 data points, and the

Table 12. The 200 data per label testing result of the 5-feature dataset.

K	Distance Matrix	Results
3	Euclidean	• The letter H often predicts the letters R and U. • The letter D often predicts the letter L.
	Manhattan	• The letter H is not detected. • The letter K often predicts the letter P. • The letter T often predicts the letter Z.
	Chebyshev	• The letter U often predicts the letter R. • The letter H often predicts the letter R. • The letter U is difficult to detect. • The letter A often predicts the letters E and D.
5	Euclidean	• The letter H often predicts the letters R and U.
	Manhattan	• The letter H is not detected. • The letter K often predicts the letter P.
	Chebyshev	• The letter Z often predicts the letter X.
7	Euclidean	• The letter H often predicts the letters R and U.
	Manhattan	• The letter H often predicts the letters R and U.
	Chebyshev	• Good detection for all letters

model with 5 features of the Chebyshev dataset with K=7 which also includes 200 data points. Both models have confusion matrix mapping which is shown in Figs. 19-20.

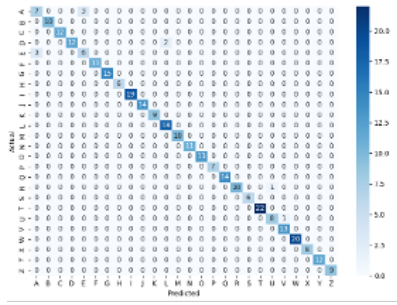

Fig. 19. The confusion matrix for Manhattan with K=3 includes 200 data points.

Based on the test results through the

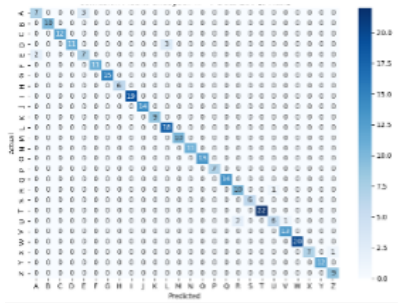

Fig. 20. The confusion matrix for Chebyshev with K=7 includes 200 data points.

variation of features and confusion matrix obtained, the model with a dataset of 4 features and 200 data types did not work well. However, there is one model that can detect everything well, even though some letters can only be detected from a distance. Across all data types from 50 to 200, the Manhattan model with K=3 shows 98% accuracy on 200 data types, making it the best model on the 4-feature dataset.

On the other hand, the test results displayed in the 200 data table with a dataset of 5 features show that the model can detect everything from a distance near or far from the camera, as long as this distance is considered ideal. However, this model is still unstable in predicting some letters. Among all trials from 50 to 200 data types, the K-NN model using the Chebyshev metric with K=7 on 200 data per label showed the best results, with a model accuracy of 97%. Although the accuracy of this model is lower than some other models, it can predict labels correctly and is more stable.

These results also show that using the K-NN algorithm with a 5-feature model has better prediction accuracy compared to previous research which also used other conventional algorithms. This is proven through the accuracy comparison shown in Table 13.

Table 13. Comparison of the related works in Sign Language.

Method	Accuracy
KNN 5-feature model	98%
KNN Using LDA HOG Extraction	72%
Random Forest	97%
Logistic Regression	96%

6.5 Comparison between model and objective testing

The results of models tested with test data assess how well the model works in predicting the labels, but these tests do not provide a good assessment when applying the real model. Therefore, it is necessary to test the model in real time. The entire model is being tested directly using the camera. The real-time test results obtained the best model from the 4-feature dataset and the 5-feature dataset, where the two models were compared to select one as the best model to be applied to simple applications and tested on other subjects.

6.5.1 Comparison between 4 and 5 dataset features model

The comparison is carried out subjectively, namely choosing a model from the results of real-time comparison trials where the model obtained by the 4-feature dataset is Manhattan with K=3 includes 200 data points, and the model obtained by the 5-feature dataset produces Chebyshev with K=7 includes 200 data points as the best model. Real-time test comparison is shown in Table 14.

The results presented in Table 14 can be compared through Table 15.

The comparison obtained shows that Chebyshev obtained the best model from a dataset of 5 features, where the duplicate distance feature can trigger stability in predicting the label.

Table 14. Comparison of the related works in Sign Language.

Letter	Manhattan			Chebyshev		
	True(%)	Labelling	False(%)	True(%)	Labelling	False(%)
A	100			100		
B	100			100		
C	100			100		
D	100			100		
E	100			100		
F	100			100		
G	100			100		
H	66,67	U	33,33	100		
I	100			100		
J	100			100		
K	100			85,71	P	14,29
L	100			100		
M	66,67	S	33,33	100		
N	100			71,43	U	28,57
O	100			100		
P	100			100		
Q	100			100		
R	100			100		
S	100			100		
T	100			100		
U	100			100		
V	100			100		
W	100			100		
X	100			100		
Y	100			100		
Z	100			100		

Table 15. Comparison between 4 and 5-feature dataset models.

Comparison	4 Features	5 Features
Data type	Data 200	Data 700
Model	Manhattan K = 3	Chebyshev K = 7
Low probability	Letter H and M	Letters K and N
Detection Capability	All letters	All letters
Letters distance detection	letters A and R are not detected	All letters can detect

6.5.2 Objective testing result

This testing involves general people as testers of the selected model. This model was tested on 10 participants, which showed different probability values. The results from 5 male and 5 female participants are presented in Tables 16-17..

In the test results on female subjects, several labels experienced a decrease in probability values. Some were even undetectable. Results are influenced by hand shape, lighting, similar shapes to other letters, and sensitivity of the mediapipe. The following is a real-time test table with male subjects.

Testing results on male subjects experienced things that were not much different from testing previous subjects. The result was related to the same thing, namely, labels similar to other letters, influenced by hand shape and the sensitivity of hand detection from the mediapipe library. So, the hand index has a significant change in detection.

Table 16. Female hand detection testing result.

Letters	F1 (%)	F2 (%)	F3 (%)	F4 (%)	F5 (%)
A	100	100	100	100	71,43
B	100	100	100	100	100
C	100	100	100	100	100
D	85,71	100	71,43	71,43	71,43
E	85,71	71,43	71,43	85,71	57,14
F	100	100	100	100	100
G	100	100	100	100	71,43
H	100	100	100	100	57,14
I	100	100	100	100	100
J	71,43	100	88,71	100	85,71
K	100	71,43	100	100	100
L	71,43	100	100	57,14	71,43
M	100	100	100	71,43	100
N	100	100	100	100	100
O	100	100	100	100	100
P	100	100	100	100	57,14
Q	100	85,71	100	100	100
R	71,43	100	100	71,43	100
S	100	100	100	100	100
T	100	100	100	100	100
U	85,71	100	71,43	100	57,14
V	85,71	100	100	100	100
W	100	100	100	100	100
X	100	71,43	100	No detect	71,43
Y	100	100	100	100	100
Z	100	100	100	100	100

Table 17. Male hand detection testing result.

Letters	M1 (%)	M2 (%)	M3 (%)	M4 (%)	M5 (%)
A	57,14	57,14	100	100	85,71
B	100	100	100	100	100
C	100	100	100	100	100
D	57,14	57,14	100	100	100
E	57,14	No detect	57,14	100	No detect
F	100	100	100	100	100
G	100	100	100	100	100
H	100	100	100	85,71	71,43
I	100	100	100	100	100
J	100	100	71,43	100	71,43
K	100	100	100	100	100
L	100	100	100	100	85,71
M	100	100	100	100	100
N	100	100	100	100	100
O	100	100	100	100	100
P	100	100	100	100	100
Q	100	100	100	100	85,71
R	71,43	57,14	57,14	85,71	100
S	100	100	100	100	100
T	100	100	100	100	100
U					
V	100	100	100	100	100
W	100	100	100	100	100
X					
Y	100	100	100	100	100
Z					

6.6 SIBI performance

The model application is used to help display letters that match the image displayed so that if the hand gesture is different from the image's shape, it will give a red indicator, which means the hand gesture does not match the image displayed. The following is an example of a test shown in Fig. 21.

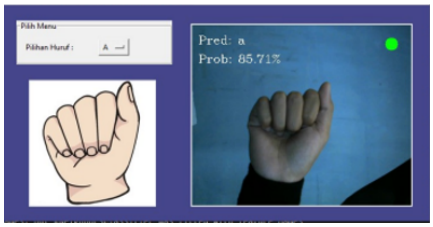

Fig. 21. True detection of the letter A.

The image above shows the test for the hand signal A, which is shown in the dropdown list. Selecting the letter A, which then displays the image A, so that if the hand signal detected by the camera is the letter A, it will display a green indicator. If it is not appropriate, it will display red. In addition, it also displays the probability corresponding to the signal it detects. Error detection in the system is shown in Fig. 22, where this image shows the choice of letter A, but the hand gesture forms the letter B, which means it does not match the choice of letter and gesture shown.

Fig. 22. False detection of the letter A.

7. Conclusion

SIBI produced results in which 4-feature obtained the best model for 200 data types, Manhattan distance, and K=3, with the ability to detect overall. However, the decreasing probability value for the letters H and M and A and R can be detected over a long distance. The 5-feature obtained stable detection results; they could detect at the ideal distance even though the probability value for the letter K and the letter N decreased, namely the Chebyshev model with K=7 from 200 data types. So, in conclusion, the classification of 5 features obtained better results than the classification of 4 features. The detection ability of the 5-feature is more stable. After all, it uses duplicate features where the 4 feature has the same distance as the 5 features, with the position of the 5 feature being able to trigger and strengthen stability in letter detection. Based on comparing real-time detection tests and observation datasets with similar value properties between features.

The model chosen is Chebyshev with K=7, 200 data types from a dataset of 5 features applied to the GUI, where this application makes it easier to carry out tests on ten people, consisting of five men and five women. The final test results show that each label's probability value depends on the hand's shape. Apart from that, other factors, such as light and the sensitivity of hand detection by the media pipe library, greatly influence the probability value of the character. At first, the model was very good from the author's point of view, but the model could be said to be not good in certain people's trials.

Based on the research experience, the authors gained valuable insights into balancing accuracy, computational efficiency, and stability, which are critical factors in the successful deployment of machine learning systems in real-world applications.

For future work, we will focus on enhancing the real-time classification of SIBI by improving its robustness against outliers, optimizing performance on larger datasets, and integrating advanced preprocessing techniques to handle diverse sign variations more effectively.

References

[1] L. L. Wijaya, Bahasa Isyarat Indonesia sebagai Panduan Kehidupan bagi Tuli, Feb. 2019.

[2] D. Maryani and R. R. E. Nainggolan, Pemberdayaan Masyarakat. Deepublish, Oct. 2019.

[3] D. R. R. M.Pd, Sistem Komunikasi Anak Dengan Hambatan Pendengaran. Deepublish, Aug. 2021.

[4] R. B. Widodo, Machine Learning Metode k-Nearest NeightBors Klasifikasi Angka Bahasa Isyarat. Media Nusa Creative (MNC Publishing), Aug. 2022.

[5] F. Mutia, AKSES, INFORMASI DAN DISABILITAS. Airlangga University Press, Jul. 2023.

[6] Y. M.Kom, ST, Data Mining. CV Jejak (Jejak Publisher), Dec. 2022.

[7] P. Yugopuspito, I. M. Murwantara, and J. Sean, Mobile Sign Language Recognition for Bahasa Indonesia using Convolutional Neural Network, in Proceedings of the 16th International Conference on Advances in Mobile Computing and Multimedia. Yogyakarta Indonesia: ACM, 2018:84-91.

[8] V. R. Tiwari, Developments in K.K.K.D. Tree and KNN Searches, International Journal of Computer Applications, 2023;185(17):17-23.

[9] R. Syafaruddin, A. A. Ilham, and I. Nurtanio, Indonesian Sign Language

Letter Interpreter Application Using Leap Motion Control based on Naive Bayes Classifier, IOP Conference Series: Materials Science and Engineering, 2019;676(1):012012.

[10] N. Afifah and H. Nughroho, Deteksi Fitur Huruf Sistem Isyarat Bahasa Indonesia (SIBI) Menggunakan Metode Chain Code, Des 2022. 56.

[11] H. M. Putri and W. Fuadi, Pendeteksian Bahasa Isyarat Indonesia Secara Real-Time Menggunakan Long Short-Term Memory (LSTM), 2021.

[12] I. Suyudi, S. Sudadio, and S. Suherman, Pengenalan Bahasa Isyarat Indonesia menggunakan Mediapipe dengan Model Random Forest dan Multinomial Logistic Regression (Introduction to Indonesian Sign Language Using Mediapipe With Random Forest Models and Multinomial Logistic Regression), 2022;1(1):2022.

[13] M. E. Al Rivan, H. Irsyad, K. Kevin, and A. T. Narta, Implementasi LDA pada fitur HOG untuk Klasifikasi ASL Menggunakan K-NN, JATISI (Jurnal Teknik Informatika dan Sistem Informasi), 2020;7(2):214-25.

[14] N. A. Hasma, F. Arnia, R. Muharar, and M. K. Muchamad, Pengenalan Gerakan Isyarat Bahasa Indonesia Menggunakan Algoritma SURF dan K- Nearest Neighbor, 2022;7(1):50-4.

[15] S. V. Sonekar, H. Dhoke, V. Mate, S. Dhewle, and Marshneel Patil, Real-Time Sign Language Identification using KNN: A Machine Learning Approach, IEEE 11th International Conference on Emerging Trends in Engineering & Technology-Signal and Information Processing (ICETET-SIP), Jun. 2023.

[16] R. Dullo, Y. Kamble, V. Suryawanshi, S. Mishra, and P. Patilm, A Study on Real-Time Sign Language Identification Using KNN, Madya Bharti, Humanities and Social Sciences, 2024;83(22):111-20.

[17] 7 LCD Display Joy-IT. Available: https://joy-it.net/en/products/RB-LCD-7-2.

[18] T. Cover and P. Hart, Nearest neighbor pattern classification, in IEEE Transactions on Information Theory, 1967;13(1):21-7.

[19] Devroye, Luc; Gyorfi, Laszlo; Lugosi, Gabor (1996). A probabilistic theory of pattern recognition. Springer. ISBN 978-0-3879-4618-4.

[20] W. Budiharto, Machine learning dan computational intelligence. Yogyakarta: Penerbit ANDI.

[21] C. Fu and J. Yang, Granular Classification for Imbalanced Datasets: A Minkowski Distance-Based Method, Algorithms, 2021;14(2):54.

[22] S. J. Russell, P. Norvig, and E. Davis, Artificial Intelligence: A Modern Approach. Prentice Hall, 2010.

[23] I. Iswanto, T. Tulus, and P. Sihombing, Comparison of Distance Models on K-Nearest Neighbor Algorithm in Stroke Disease Detection, Applied Technology and Computing Science Journal, 2021;4(1):63-8.

[24] Y. Heryadi and T. Wahyono, Machine Learning (Konsep dan Implementasi). Yogyakarta: Gava Media, 2020. 57.

[25] U. Fadlilah, A. K. Mahamad, and B. Handaga, The Development of Android for Indonesian Sign Language Using Tensorflow Lite and CNN: An Initial Study, Journal of Physics: Conference Series, 2021;1858(1):012085.

[26] S. Nur Budiman, S. Lestanti, S. Marselius Evvandri, and R. Kartika Putri, Pengenalan Gestur Gerakan Jari Untuk Mengontrol Volume Di Komputer Menggunakan Library Opencv dan Mediapipe,

Antivirus: Jurnal Ilmiah Teknik Informatika, 2022;16(2):223-32.

[27] Suharjito, N. Thiracitta, and H. Gunawan, SIBI Sign Language Recognition Using Convolutional Neural Network Combined with Transfer Learning and non-trainable Parameters, Procedia Computer Science, 2021;179:72-80.

[28] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, MediaPipe Hands: On-device Real-time 58 Hand Tracking, Jun. 2020, Available: http://arxiv.org/abs/2006.10214.

[29] M. Nishom, Perbandingan Akurasi Euclidean Distance, Minkowski Distance, dan Manhattan Distance pada Algoritma K-Means Clustering berbasis Chi-Square, Jurnal Informatika: Jurnal Pengembangan IT, 2019;4(1):20-4.