Journal of Engineering Science and Technology
Special Issue, Vol. 18, No. 4 (2023) 140 - 152

TELEMEDICINE SIGN LANGUAGE CLASSIFICATION FOR COVID-19 PATIENTS WITH DISABILITY BASED ON LSTM MODEL

MARIA SERAPHINA ASTRIANI*, MARCELL ALVIANTO

Computer Science Department, School of Computing and Creative Arts, Bina Nusantara University, Jl. K. H. Syahdan No. 9, Kemanggisan, Palmerah Jakarta, Indonesia
*Corresponding Author: seraphina@binus.ac.id

Abstract

Sign language is an important part of disabled people used for their communication in daily lives. Due to a lack of understanding in terms of sign language for other people, especially for the communication between medical personnel and disabled patients, they need an interpreter to communicate well. The authors decided to make telemedicine based on a web application integrated with Machine Learning that is capable of classifying sign language based on Indonesian Sign Language or BISINDO data that will focus on COVID-19. Machine Learning was developed with Deep Learning based on multiple Long Short-Term Memory layers and multiple Dense layers to create the model. The model provides users to classify 11 sign languages and successfully achieved 99% accuracy of Training data, 98% accuracy of Testing data, and 90% accuracy on real-time classification. The results show that LSTM architecture is able to classify sign languages, especially for COVID-19 terms.

Keywords: COVID-19, LSTM, Machine learning, Sign language, Telemedicine.

1. Introduction

Disabilities are conditions where the body or mind which makes them limit their activities and having interaction around them would be difficult. According to WHO World Report on Disability, "There are around 15% or one billion people that suffer disabilities in this world, and around 110 – 190 million experience significant disabilities [1]. All disability conditions are congenital and adventitious from birth. Deaf and mute are conditions that need a special way to communicate with other people. There are two-way communications such as verbal, and non-verbal communication. Verbal communication that only uses one word or more. Meanwhile non-verbal communication is communication beyond spoken or written words, which can be in the form of body language, facial expressions, sign language, etc. [2]. One of the sign languages in Indonesia is called BISINDO, sign language that occurs naturally in Indonesian culture and is used in everyday life [3].

Virus is a part of this world. On 31 December 2019, a new virus called Coronavirus disease 2019 called COVID-19 discovered in Wuhan city, China and spread to other countries quickly [4]. Due to the COVID-19, the most important thing in our life is people's health conditions. Technology has an important role in this pandemic, which impacts our daily lives. Telemedicine is one of the implementations of technology in the health field. Telemedicine combines the technology with medical service. According to the Law of Health Minister Number 20 of 2019, stated that "Telemedicine is a place for medical services for any patient needs in terms of their health conditions by providing them with a long-distance communication between patients and medical personnel" [5].

Machine Learning plays a big role in technologies in this era. Machine Learning can be used to classify and visualise sign language for who needs it. In Indonesia, disabled people (deaf and mute) need more accessibility support systems regarding their limitations in terms of listening and communication, which cause communication problems to communicate with medical personnel. Deaf and mute people may be helped by "sign language as long as there is an interpreter between them" [6]. Unfortunately, not all medical personnel and hospitals had the ability to understand or provide human resources as an interpreter, and it also occurs inside telemedicine service between disabled patients and medical personnel.

The proposed model aims to classify BISINDO signs and show the high accuracy result. It is expected to help someone that does not understand sign language to understand and help patients to communicate with doctors.

2. Related Works

The process to classify sign language is difficult to make it flawless. A lot of work has been done to classify sign language, because of various and similar signs which make it difficult to classify. This section shows some related work from other researchers. Li et al. [7] have developed to identify American Sign Language and Arabic numerals by using CNN-LSTM architectures with 95.52% for recognition and 93.3% for translation. These results are promising, but the challenge appears when the method needs to recognize the gesture in real-time. Sonare et al. [8] developed by using CNN-LSTM architecture with American Sign Language data and the accuracy results by using the baseline of LSTM is 86.6%, 89.2% by using the baseline of CNN, and can even achieve 90.1% when the approaches are

combined. Adhikary et al. [9] do some comparison on classification algorithms to recognize Indian Sign Language, where Decision Tree algorithm with 83.7% of accuracy, Random Forest with 97.4% of accuracy and Gradient Boosting with 95.6% of accuracy. The researchers identified this model's limitation as its inability to recognize a sufficient number of signs. Fadillah et al. [10] develop to recognize BISINDO alphabetic sign from American Sign Language with transfer learning and CNN, which receive the best result 30.63% on Train accuracy 30.66% on Validation accuracy and 30% in Testing accuracy. Susanty et al. [10] developed the recognition of the alphabet from the BISINDO dataset by implementing two types of CNN architectures (LeNet-5 and AlexNet). LeNet-5 produced lower accuracy results compared with AlexNet. AlexNet can achieve 76% accuracy, but it is only able to predict around 60% in real-time video. The researchers stated that these approaches still cannot receive higher accuracy.

3. Proposed Methodology

This section consists of different processes to create models and achieve the best results. Figure 1 shows the block diagram of the model development process.

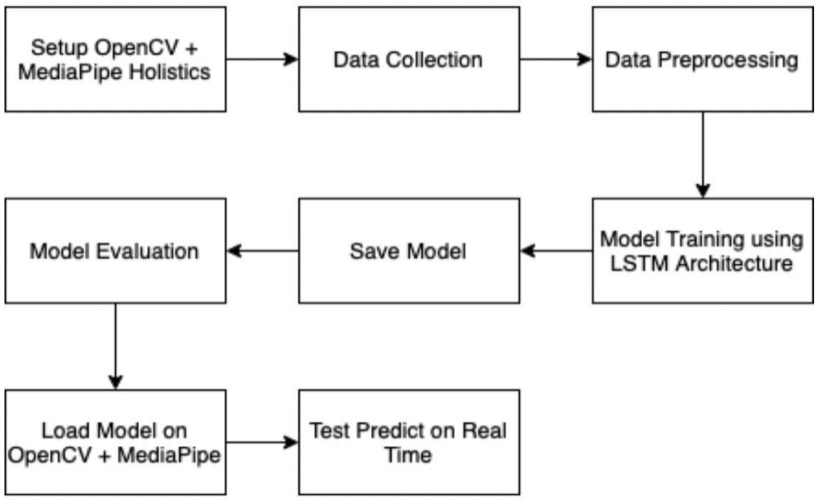

Fig. 1. Block diagram of model development.

3.1. Setup OpenCV and MediaPipe Holistics

OpenCV was needed to collect the data by using a webcam. MediaPipe Holistics was implemented inside OpenCV. The data of Sign language will be extracted depending on the key points from MediaPipe Holistics detection. It provides several parts of key points detection in real time which consists of Pose Landmark, Face landmark, Left-Hand Landmark and Right-Hand Landmark. The author used the key points that were extracted from MediaPipe Holistics with 166 key points.

3.2. Data collection and dataset

Data collection is a basic step of creating a model that is based on needs. There is no public dataset available in BISINDO (COVID-19 terms) that authors need in

this research. The data used in this research is already validated by the verified organization. By creating a path of the data based on different actions, the folder will consist of 11 labelled words (terms), with 30 videos worth of data each action and 30 frames that contain extracted key point values from MediaPipe Holistic detection. The data was collected through a webcam and each frame saved as NumPy Array format that contains 1662 key points. The words sign include "No Action", " Doctor (*Dokter*)", "Cough (*Batuk*)", "Flu (*Pilek*)", "Dizzy (*Pusing*)", "Nausea (*Mual*)", "Mask (*Masker*)", "Medicine (*Obat*)", "Fever (*Demam*)", "Sick (*Sakit*)", and "Sneeze (*Bersin*)". Figure 2 indicates the static images of BISINDO Signs that implement inside the model.

Fig. 2. Static images of each sign language data.

The data was shaped in NumPy array format by loading several files and shapes of the data that were going to be processed inside the model. The shapes consist of 330 (330 videos, 30 for each sign and 11 signs), 30 (frames of each data), and 1662 (total key points).

3.3. Data pre-processing

The collected data will be processed into the next step where it will be labelled and separated into training, validation, and testing parts. Figure 3 indicates the label (by using number) of each sign.

```
====== Label into Numbers ========
label_map :
 {'no_action': 0, 'dokter': 1, 'batuk': 2, 'pilek': 3, 'pusing': 4, 'mual': 5, 'masker': 6, 'obat': 7, 'demam': 8, 'sakit': 9, 'bersin': 10}
```

Fig. 3. Label of signs.

By looping through all the frames, the data split into three parts, 56% for training, 14% for validation, and 30% for testing. For splitting parts, by using stratification to separate equally on each label. Figure 4 indicates the distribution of data on each label after split, and data that is going to be processed for training, validation, and testing.

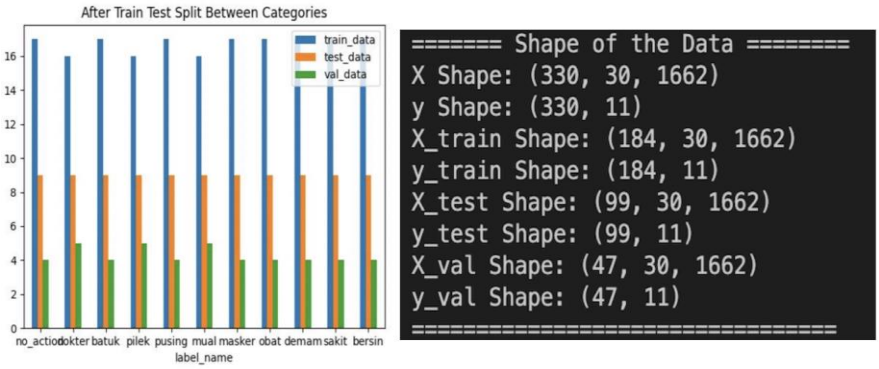

Fig. 4. Results of data pre-processing.

3.4. Model architectures

LSTM architecture where it is capable and suitable for learning and remembering long sequences data, which is often used for sequence labelling or sequence classification [11-15]. The model used Stacked LSTM architecture which means having a multiple layer of LSTM. This technique has the benefit of making it deeper and accurate. The first layer of stacked LSTM will provide a sequence output not only a single value output for the next layer [16]. The architecture of the model is shown in Table 1.

Table 1. Model architectures on each layer.

No	Layer Name	Activation
1	LSTM 1	ReLU
2	LSTM 2	ReLU
3	LSTM 3	ReLU
4	Dense 1	ReLU
5	Dense 2	ReLU
6	Dense 3	Softmax

Below are the explanations of the model by using LSTM and Dense layers:

- The first LSTM (LSTM 1) layer has 64 units with input shape (30, 1662) and the output shape (30, 64). This layer uses ReLU activation.
- The output from the first LSTM layer will become the input for the second layer. The second uses the LSTM (LSTM 2) layer and it has 128 units.
- The third LSTM (LSTM 3) layer (64 units) uses ReLU activation, and the output goes to the Dense layer.
- Dense layers use fully connected layers. The first Dense (Dense 1) layer has 64 activated neurons and ReLU activation.
- The second Dense (Dense 2) layer uses 32 activated neurons and uses the same activation like in Dense 1.
- The third Dense (Dense 3) layer uses Softmax activation. This layer will be the actions layer. It indicates the shape of the terms, which means 11 neural network units for 11 signs.

Table 2 represents parameters inside model, where the model compile with Adam for the Optimizer, Categorical Crossentropy for the Loss function, and Categorical accuracy for the metrics, with Epoch = 130, and the model will train wit validation data that already split before on data pre-processing. The Epoch stopped at 130 because the authors found the results of accuracy already high and the loss already low.

Table 2. Model parameters.

Model Architecture		
LSTM	3 Layers	Activation Function: ReLU
Dense	3 Layers	Activation Function: ReLU and Softmax
Epoch	130	
Optimizer	Adam	
Loss	Categorical Crossentropy	
Metric	Categorical Accuracy	
Validation Data	(x_val, y_val)	

3.5. Model evaluation and web application integration

After training finishes, the model will be evaluated by the author by using two performance metrics to see how well the model classifies each sign. The performance evaluation metrics consist of accuracy (ratio of true positive and true negative) and Confusion Matrix (measurement of the output) [17]. If the accuracy results using Training data and Testing data show good results (generally the accuracy shows above or equal to 90%), then the model will be tested for real-time classification. The model will be integrated in telemedicine if the real-time accuracy results show a minimum of 90%. This telemedicine was developed based on a web application to classify sign language based on BISINDO data that will focus on COVID-19.

4. Results and Discussion

This section shows the results of the model which consist of Training accuracy, Validation accuracy, and Loss curves. When developing models, technical elements including activation functions, epochs, precision, and coding are used. These elements are derived from works by Gupta [18], Sharma [19], Kumar [20], and Brownlee [21]. The measurement of the model uses accuracy because the dataset was balanced.

4.1. Training and validation curve on categorical accuracy

Figure 5 indicates the learning performances change over time. It represents the results of the learning curve of the model based on Training and Validation on Categorical accuracy. This curve indicates how well the model generalises the data. At first the model starts from 0.0543 on Train Categorical accuracy and on Validation starts from 0.1064 where at finishing point, the model reaches 0.9946 - 1 on Train an 0.9787 - 1 on Validation Categorical accuracy.

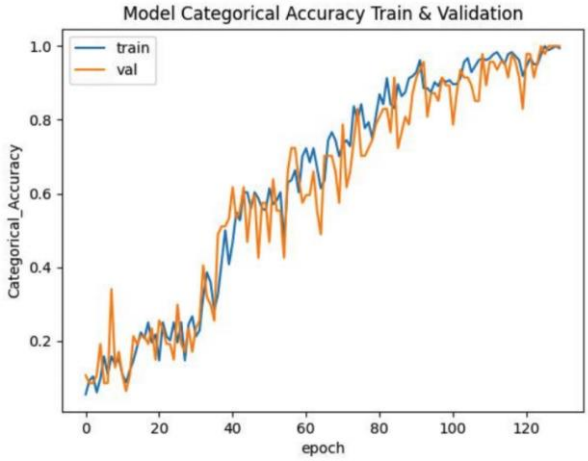

Fig. 5. Training and Validation Curves of the model.

4.2. Training and validation loss curve on categorical cross entropy

Figure 6 represents the result from training and validation. Training Loss Curve indicates how well the model fits with data where Validation Loss Curve indicates how well the model fits with new data. The model at first has Loss around 2.4678 and Validation Loss around 2.4043. The loss decreases slowly over time where on the last training. It showed the Loss around 0.0122 and Validation loss around 0.0074, with a small gap between them.

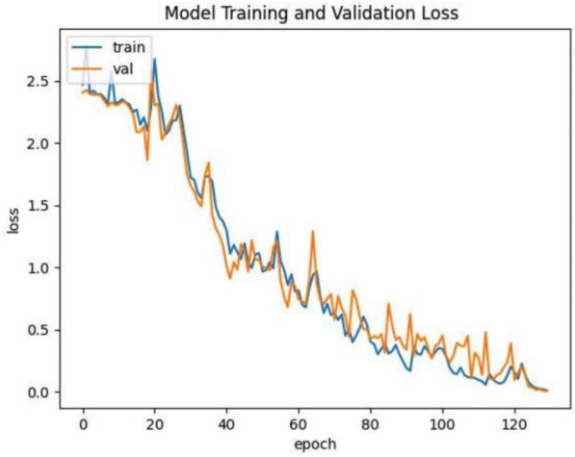

Fig. 6. Training and Validation Loss Curves of the model.

4.3. Performance evaluation of the model

Table 3 represents the accuracy of each label on Train data. The results show the calculation of accuracy of each label, where almost all of the labels reached 100%, but not with 2 labels which are "No Action" and "Sick (*Sakit*)" with accuracy 99%. The average of the results represents how well the model classifies data based on accuracy where it achieved 99% for all label classification.

Table 3. Results on train data.

Label	Accuracy
No Action	99%
Doctor (*Dokter)*	100%
Cough (*Batuk)*	100%
Flu (*Pilek)*	100%
Dizzy (*Pusing)*	100%
Nausea (*Mual)*	100%
Mask (*Masker)*	100%
Medicine (*Obat)*	100%
Fever (*Demam*)	100%
Sick (*Sakit)*	99%
Sneeze (*Bersin)*	100%
Average	**99%**

Testing the data is important to check how well the model is able to classify each label on the data that has not been recognized by the model in the Training process. Testing data that was used are data that already separate on pre-processing. Table 4 represents the accuracy on each label on Test Data. It shows that for 5 different signs that consists of "Nausea (*Mual*)", "Mask (*Masker*)", "Medicine (*Obat*)", "Fever (*Demam*)", and "Sick (*Sakit*)", reached 100%, where for "Sneeze (*Bersin*)" and "Doctor (*Dokter*)" signs reached 98%. "Dizzy (*Pusing*)" has 97% accuracy, and the lowest results show on "Cough (*Batuk*)", and "Flu (*Pilek*)" signs, with the results only 96%. The average accuracy of all labels is 98% for all label classification.

Table 4. Results on Test data.

Label	Accuracy
No Action	99%
Doctor (*Dokter)*	98%
Cough (*Batuk)*	96%
Flu (*Pilek)*	96%
Dizzy (*Pusing)*	97%
Nausea (*Mual)*	100%
Mask (*Masker)*	100%
Medicine (*Obat)*	100%
Fever (*Demam*)	100%
Sick (*Sakit)*	100%
Sneeze (*Bersin)*	98%
Average	**98%**

The authors implement the model to do classification with real-time data because the authors want to evaluate how well the model classifies the real-time data especially for telemedicine implementation. Figure 7 illustrates the result's example of "Fever (*Demam*)" real-time classification. The authors did some

experiments to find out probabilities to classify the label for each label in real-time. The experiments were conducted by repeating 10 times on each sign to determine the accuracy. The average accuracy on real-time classification is around 90% for all labels. The detailed information can be found in Table 5.

Fig. 7. "Fever (*Demam*)" real-time classification.

Table 5. Results on real-time classification.

Label	Repeat	Success	Accuracy
No Action	10	10	100%
Doctor (*Dokter)*	10	10	100%
Cough (*Batuk)*	10	10	100%
Flu (*Pilek)*	10	10	100%
Dizzy (*Pusing)*	10	8	80%
Nausea (*Mual)*	10	10	100%
Mask (*Masker)*	10	10	100%
Medicine (*Obat)*	10	10	100%
Fever (*Demam*)	10	8	80%
Sick (*Sakit)*	10	7	70%
Sneeze (*Bersin)*	10	7	70%
Average Accuracy			**90%**

4.4. Telemedicine based web application

The model has been tested for real-time classification and then implemented in a web-based telemedicine application. Figure 8 is a user interface of the telemedicine home page. This page explains the online doctor consultation service can also be performed for disabled people (deaf and mute) because it has a feature to detect sign language based on BISINDO. The list of sign languages that can be detected by telemedicine is displayed in the "BISINDO Sign Language" section. The sign language telemedicine feature focuses on COVID-19's terms. Disabled patients can consult with a doctor after signing up and logging in to telemedicine. The user interface for doctor-patient consultations is made more user-friendly by incorporating video call concepts. Because telemedicine has additional features that can recognize sign language based on BISINDO for COVID-19 terms, the user interface has additional information on the top

left of the screen and a "skeleton" on the patient. If the patient uses sign language and is successfully classified by the model, the word will be highlighted. Figure 9 is the display result for the results of the "No Action" classification.

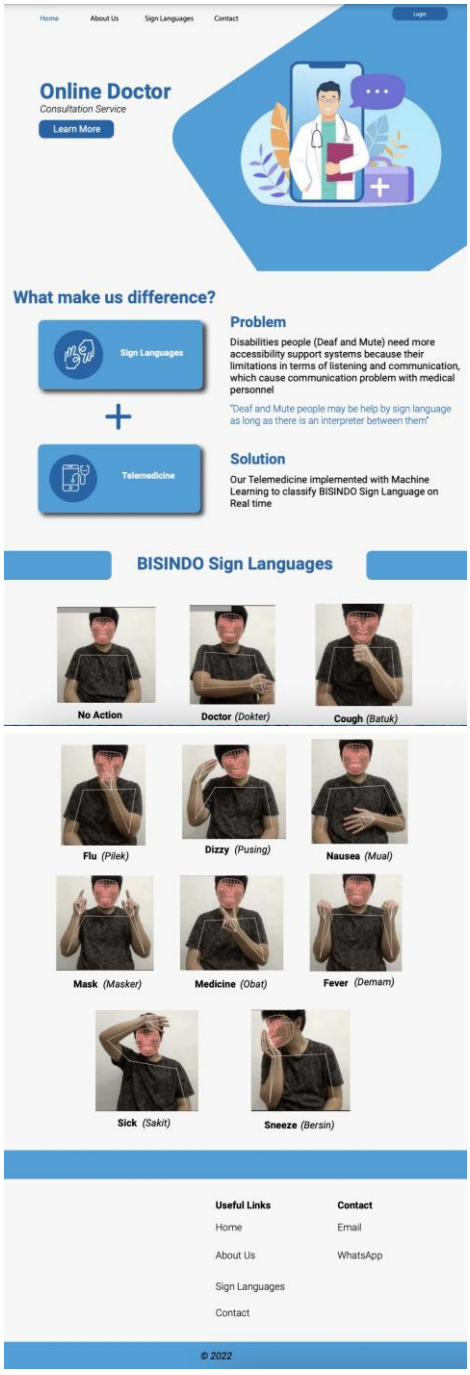

Fig. 8. Telemedicine user interface - home page.

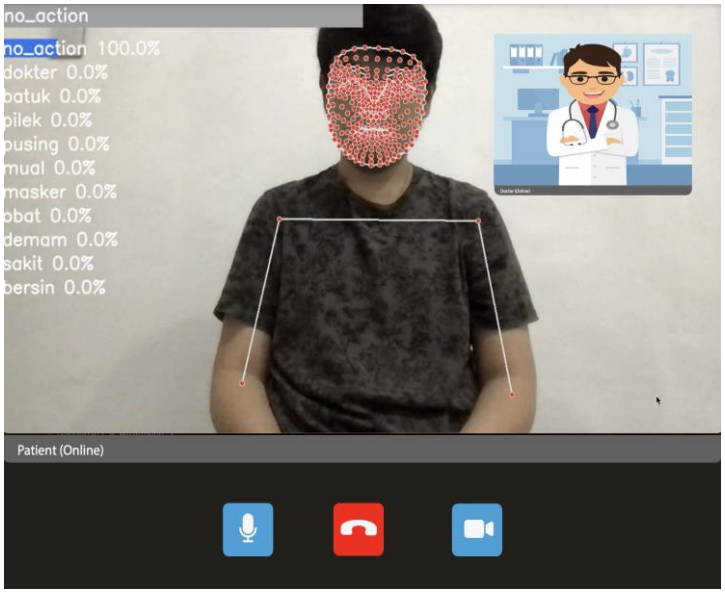

Fig. 9. Telemedicine user interface - patient during consultation.

5. Conclusion and Recommendation

The authors trained the model with key points dataset extracted from MediaPipe that indicate each label. The data consist of 330 data in total, 30 on each sign (11 signs in total) and using LSTM architecture in the model. The results show that LSTM Architecture is suitable for classifying sign languages especially for COVID-19 terms based on BISINDO. There are 11 signs, with different kinds of movements, some movements are similar, and some are not. From the real-time classification, "Dizzy (*Pusing*)", "Sick (*Sakit*)", and "Sneeze (*Bersin*)" have similar sign language by using the palm of the hand to indicate it and the model has difficulty in classification them. It can be concluded that the machine learning model successfully classifies 11 signs by achieving 99% accuracy of Training data, 98% accuracy of Testing data, and 90% accuracy on real-time classification.

The model is sometimes mistaken to classify similar signs. The recommendation to improve the accuracy performance of the model is to add another LSTM layer and the number of train data for each label so the model can learn more and better to do classification on different sign language movements.

Abbreviations

BISINDO	*Bahasa Isyarat Indonesia* or Indonesian Sign Language
CNN	Convolutional Neural Network
CNNLSTM	Convolutional Neural Network Long Short-Term Memory
COVID-19	Coronavirus Disease of 2019
LSTM	Long Short-Term Memory
ReLu	Rectified Linear Unit
WHO	World Health Organization

References

1. WHO. World report on disability. Retrieved August 31, 2022, from https://www.who.int/news-room/fact-sheets.
2. DPP PPDI. (2020). Pola komunikasi tuna rungu di dewan Perkumpulan Penyandang Disabilitas Indonesia. Retrieved February 17, 2022, from https://ppdi.or.id/jurnal-pola-komunikasi-tuna-rungu-di-dewan-perkumpulan-penyandang-disabilitas-indonesia/?__cf_chl_f_tk=eG2rYBrygiThHTn4N7xlSvWkDM_1n.Yk0ej5jw9rdDc-1642313580-0-gaNycGzNB6U.
3. Agustin, S. (2022). Peran bahasa isyarat untuk kelancaran komunikasi anak. Retrieved February 17, 2022, from https://www.alodokter.com/peran-bahasa-isyarat-bagi-penderita-tuna-rungu-dan-anak-anak.
4. CDC (2021). Basics of COVID-19. Retrieved March 10, 2022, from https://www.cdc.gov/coronavirus/2019-ncov/your-health/about-covid-19/basics- covid-19.html
5. Harbani, R.I. (2021). Telemedicine diluncurkan di DKI, apa bedanya dengan konsul Kesehatan online?. Retrieved February 17, 2022, from https://www.detik.com/edu/detikpedia/d-5633264/telemedicine-diluncurkan-di-dki-apa-bedanya-dengan-konsul-kesehatan-online
6. RS PKU Jogja. (2017). Aksesibilitas bagi penyandang disabilitas di rumah sakit. Retrieved August 31, 2022, from https://rspkujogja.com/aksesibilitas-bagi-penyandang-disabilitas-di-rumah-sakit/
7. Li, W.; Pu, H; and Wang, R. (2022). Sign language recognition based on computer vision. *Proceedings of 2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*. Dalian, China, 919-922.
8. Sonare, B.; Padgal, A.; Gaikwad, Y.; and Patil, A. (2021). Video-based sign language translation system using machine learning. *Proceedings of the 2021 2nd International Conference for Emerging Technology (INCET)*. Belagavi, India, 1-4.
9. Adhikary, S.; Talukdar, A.K.; and Sarma, K.K. (2021). A vision-based system for recognition of words used in Indian sign language using MediaPipe. *Proceeding of 2021 Sixth International Conference on Image Information Processing* (*ICIIP*). Assam, India, 390-394.
10. Susanty, M., Fadillah, R.Z.; and Irawan, A. (2022). Model penerjemah Bahasa isyarat Indonesia (BISINDO) menggunakan pendekatan transfer learning. *PETIR: Jurnal Pengkajian dan Penerapan Teknik Informatika*, 15(1).
11. Brownlee, J. (2017). Techniques to handle very long sequences with LSTMs. Retrieved February 17, 2022, from https://machinelearningmastery.com/handle-long-sequences-long-short-term-memory-recurrent-neural-networks/
12. Yu, Y.; Si, X.; Hu, C.; and Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural computation*, 31(7), 1235-1270.
13. Zha, W.; Liu, Y.; Wan, Y.; Luo, R.; Li, D.; Yang, S.; and Xu, Y. (2022). Forecasting monthly gas field production based on the CNN-LSTM model. *Energy*, 260, 124889.

14. Astriani, M. S.; Heryadi, Y.; Kusuma, G. P.; and Abdurachman, E. (2019). Long short-term memory for human fall detection based gamification on unconstraint smartphone position. *Proceedings of the 2019 International Congress on Applied Information Technology* (*AIT*). Yogyakarta, Indonesia, 1-6.
15. Moghar, A.; and Hamiche, M. (2020). Stock market prediction using LSTM recurrent neural network. *Procedia Computer Science*, 170, 1168-1173.
16. Sak, H.; Senior, A.W.; and Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. *Interspeech*, 338-342.
17. Narkhede, S. (2018). Understanding confusion matrix. Retrieved February 17, 2022, from https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62.
18. Gupta, D. (2020). Fundamentals of Deep Learning – activation functions and when to use them?. Retrieved February 17, 2022, from https://www.analyticsvidhya.com/blog/2020/01/fundamentals-deep-learning-activation-functions-when-to-use-them/.
19. Sharma, S. (2017). Epoch vs batch size vs iterations. Retrieved February 17, 2022, from https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9.
20. Kumar, A. (2022). Accuracy, precision, recall & F1-score – Python examples. Retrieved February 17, 2022, from https://vitalflux.com/accuracy-precision-recall-f1-score-python-example/.
21. Brownlee, J. (2017). Stacked long short-term memory networks. Retrieved February 17, 2022, from https://machinelearningmastery.com/stacked-long-short-term-memory-networks/.