OPEN ACCESS
ISSN 2356-5462
http://socj.telkomuniversity.ac.id/ijoict/

Intl. Journal on ICT
Vol. 9, No. 2, Dec 2023. pp. 86-95
doi: doi.org/10.21108/ijoict.v9i2.845

The Recognition of American Sign Language Using CNN with Hand Keypoint

Muhamad Asep Ridwan [1], Aradea [2], Husni Mubarok [3*]

[1,2,3] *Departmaent of Informatics Engineering, Siliwangi University*
Jl. Siliwangi No.24, Tasikmalaya, Jawa Barat, Indonesia

*Husni.Mubarok@unsil.ac.id

Abstract

Sign Language is a method used by the deaf community for their communication. In line with the advances of deep learning, researchers have widely interpreted neural networks for language recognition in recent years. Many models and hardware have been developed to help get high accuracy in language recognition, but generally, the problem of accuracy is still a concern of researchers, even the accuracy problem related to American language or American sign language (ASL) still requires further research to solve. This paper discusses a method to improve ASL recognition accuracy using Convolutional Neural Network (CNN) with hand keypoint. Pre-trained Keypoint detector is used to generate hand keypoints on the massey dataset as an input for classification in the CNN model. The results show that the accuracy of the proposed method is better than the previous studies, obtaining an accuracy of 99.1% in recognizing the 26 statistical signs of the ASL alphabet.

Keywords: *American Sign Language, Convoltional Neural Network, Hand keypoint, Massey dataset*.

I. INTRODUCTION

SIGN language is a communication method that does not use sound, but uses hand, body and lip movements to convey information. Sign language has been used as a communication system, especially by people who are deaf or hard of hearing [1]. Sign language in each country has a different sign. American sign language (ASL) is a complete visual language created through hand gestures combined with facial expressions and body gestures. ASL consists of 6000 moves. Finger spelling is used to communicate difficult-to-understand words. Finger spelling consists of 26 hand gestures that show 26 alphabets [2]. ASL is one of the sign languages that has a standard dataset and is the most widely used by researchers in sign language recognition.

One of the nuances in sign language is how often fingerspelling is used. Fingerspelling is a method of spelling words using only hand gestures. One of the reasons alphabet fingerspelling has an important role in sign language is to spell unique words such as a person's name, product brand, title, place, food, animal or plant that is not common[3]. However, to recognize and classify letters accurately, it is necessary to use sophisticated hardware such as Microsoft Kinect, so the recognition process for each letter has an important role in the recognition of sign language.

Sign language recognition has been initiated by Vogler and Mexatas [4], this work presents an approach to ASL recognition that aspires to be a solution to scalability problems. It is based on parallel HMMs (PaHMMs), which independently shows and refers to the parallel processes. Thus, the model can also be trained independently, and does not require consideration of different combinations at the time of training. However, because PaHMM is a shallow learning, it is less effective for larger vocabulary. Some researchers

Received on 23 Aug 2023. Revised on 19 Sep 2023. Accepted and Published on 16 Dec 2023.

use CNNs to solve this problem. Fox et al [5] used CNN to optimize accuracy in ASL recognition. Apart from that, the pattern recognition process must also be assisted with additional hardware or sensors. Garcia et al [6] proposed deepCNN with pre-trained GoogLeNet to obtain 72% accuracy. Lower than the method of Bheda & Radpour, 2017 [7] which proposed deepCNN without pre-training and only normalizing the dataset, getting an accuracy of 82%. Based on previous research [6] did not normalize the data and [7] used depCNN without pre-training. In this study, the introduction of ASL using CNN with pre-training and normalizing the dataset obtained an accuracy of 99.1

This study classified 2515 ASL massey images [3] which contained images of digits and letters of the alphabet, the purpose of this research was to classify them into the correct class. Pre-trained Keypoint detector is used to extract hand keypoints from the massey dataset then the hand keypoints are classified using CNN. The second part will discuss the previous research, the third part describes the proposed model, the fourth part shows the results of applying the model to case studies, and the fifth part contains conclusions and future work.

II. LITERATUR REVIEW

In recent years, Convolutional Neural Networks have been very successful in image recognition and classification problems, and have been successfully applied to sign language recognition. Various methods of sign language recognition using CNN have been studied by researchers. For example, Rao et al.,[8] proposed the recognition of Indian sign language using CNN, the recognition input is a video selfie so that people with hearing impairment can operate sign language recognition applications independently. Tao et al., [9] described the method of recognizing the American sign language alphabet using Convolutional Neural Networks (CNN) with multiview augmentation and inference fusion, from depth images captured by Microsoft Kinect. This approach can augment the original data by generating more perspectives, which makes training data more effective and reduces the potential for overfitting. Suharjito et al., [10] implemented an inflated 3D CNN model with transfer learning methods from ImageNet and Microsoft Kinect, this approach can overcome the problem of scanty data. The discussion of these related works focuses on the recognition of sign language using CNN. V. Jain et al., used Support Vector Machine (SVM) and CNN for an ASL recognition system[11].

In addition, there are several researchers who have contributed to the other sides of the domain. Specifically, those that use existing Massey datasets, such as Taskiran et al., [12] who applied the CNN + Skin Detection and Convex Hull model to the dataset from Massey University with a total of 900 data divided into 80% training data and 20% testing data. Dong et al [13] used depth-sensing technology and other tools that were incorporated into a process that had proven successfully. The developments of specially designed colour gloves are used to facilitate the recognition process and make the feature extraction step more efficient and make certain gesture units easier to identify. Garcia et al [6] utilized very basic camera technology to generate a dataset of images, with no depth or contour information, only the pixels present. Attempts to use CNN to classify images of ASL letters were successful, only with the help of the previously trained GoogLeNet architecture. Rathi et al., [14] designed sign language recognition using deepCNN, with sensitive input recognition rather than just image pixels using a camera that detects depth and contours, the process was easier by developing the profile of depth and unique motion to each sign language, using only Microsoft Kinect. Using Microsoft Kinect which has a depth sensor, can solve problems such as skin tone and lighting easily Bheda & Radpour, [7]. However, such cameras and technology are not widely accessible, and are expensive. Thus, the dataset needs to be normalized by changing the size to 200x200 pixels so it would be easy to classify using only the general CNN architecture and produces an accuracy of 82.5%. Kasapbas A et al., created an American Sign Language recognition system using CNN with two American Sign Language recognition databases [15] and [16] which produced an accuracy of 99.38 [17]. Alsharif B et al, compared five technologies for sign language recognition including AlexNet, ConvNeXt, EfficientNet, ResNet-50, and VisionTransformer which were trained and tested using an extensive dataset consisting of more than 87,000 ASL alphabet hand gesture images. The research experimental results reveal that ResNet-50 achieves an extraordinary accuracy rate of 99.98%, the highest among all models. EfficientNet obtained an accuracy level of 99.95%, ConvNeXt obtained an

accuracy of 99.51%, AlexNet obtained an accuracy of 99.50%, while VisionTransformer produced the lowest accuracy of 88.59%.[18]

The implementation of CNN with pre-trained GoogLeNet developed by Garcia et al [6] and the dataset normalization technique performed by Bheda & Radpour [7] have inspired researchers in making ASL recognition models by utilizing the pre-trained data. The differences of the pre-trained model of ASL recognition that the researcher proposed with previous studies, namely Garcia et al[6] who used the GoogLeNet architecture which has been trained previously on the 2012 ILSVRC dataset are that the researcher used a pre-trained Keypoint detector Simon et al [19]. Normalized the image by adding padding on both sides of the image so that the ratio of the image does not change when the image size is changed to 200x200 pixels. So that the accuracy of ASL recognition can be improved.

III. RESEARCH METHOD

The description of the created model has two processes, the first is image pre-processing to generate hand keypoints using a pre-trained caffemodel as an input data at the classification stage. The second stage is the process of training and testing hand keypoint data using CNN. For a complete explanation of the process, it will be explained in each sub-chapter.

A. System Architecture

The system architecture describes the stages of making the system from input to output.

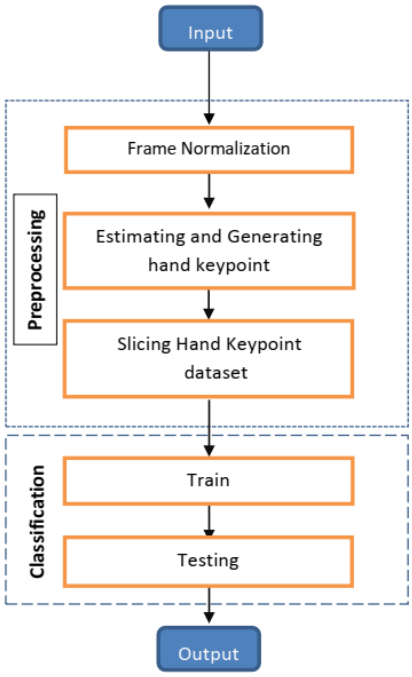


Fig. 1 System Architecture

Based on Figure 1, the flow of the recognition of the ASL system using the CNN model with hand keypoint has two stages, the first is pre-processing and the second is the classification stage. The pre-processing stage starts from the input of the raw image of the massey dataset then normalizes the image by changing the image size to 200x200 pixels and adds padding on both sides so that the ratio of the image does not change, then extracts the hand keypoint on the palm using a pre-trained caffémodel which results in the form of data array of hand keypoints. At the Slicing hand keypoint dataset stage, the process of cutting the training and testing data is carried out. After the pre-processing stage is complete, which produces an array of training data and an array of testing data. Furthermore, it is processed at the

Classification stage. At the training stage, the training process is carried out on the training data that has been cut at the previous pre-processing stage, the results of the training are in the form of a trained model. The next step is testing the data, the results are metric values of accuracy and loss value. The algorithm of ASL recognition is shown in algorithm 1.

Based on the system architecture, Table I describes the algorithm of ASL recognition using CNN with Hand Keypoint to increase accuracy in ASL recognition. The input to the system is a raw image of the massey dataset. The image is normalized so that it becomes a 200x200 pixel image with additional padding. Extraction of normalized images is to estimate and produce hand keypoints. Hand keypoints are stored as array data, then the array data is cut into testing data and training data which are used for training and testing stage.

TABEL I.
THE ALGORITHM OF ASL

The Recognition of American Sign Language Using CNN with Hand Keypoint
Input *image (dataset massey)* *Do* *Let* *//Preprocessing* *//Frame Normalization* *For each image in dataset do (Normalization)* *resize image 200x200 pixel and add padding* *end for* *//Estimating and Generating Hand keypoint* *for each image in Normalized image do (extraction)* *estimating and generating array hand keypoint with pre-trained caffemodel* *end for* *//Slicing Hand keypoint dataset* *split each class in array hand keypoint with ratio(85-15) as train set and test set* *//Classification* *for each epoch to 100* *for each batch train in train set do (train)* *training model* *end for* *for each batch test in test set do (test)* *testing model* *end for* *end for* *output* *accuracy, loss*

B. CNN Architecture

The CNN architecture describes the model used to classify massey datasets. The definition of the model is shown in Figure 2.

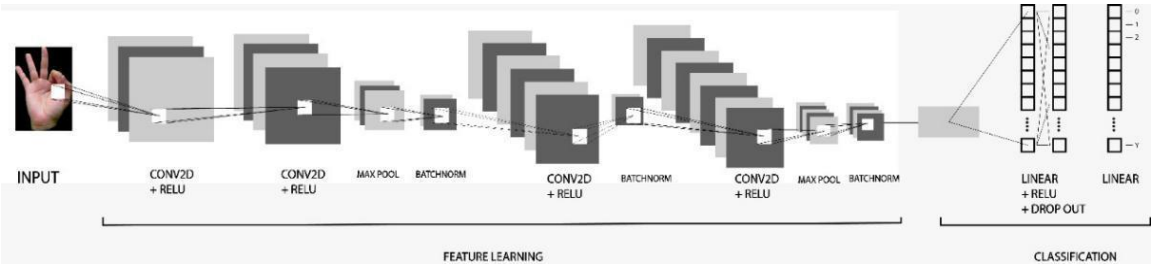


Fig. 2 CNN Architecture

Based on the model, there are two layers, the first is the feature extraction layer, which is an image feature extraction layer by layer using four convolutional layers that are integrated with the non-linear ReLu function. ReLu can help the computing process to be faster than tanh. In addition, two layers of maxpool are added after the second and fourth convolution layers to reduce image dimensions to speed up the computational process. Batch normalization layer is applied to the second, third and fourth convolution layers to make training on CNN more efficient.

nn.Conv2d:

- in_channels (int) — Number of channels in the input image (for grayscale image: 1 and for RGB image: 3)
- out_channels (int) — Number of channels produced by the convolution. Here we have defined 5, 10, 20, 40. The output of every out_channes will be the input (in_channels) for the next convolutional layer.
- kernel_size (int or tuple) — Size of the convolving kernel. Here, we have chosen 3.
- stride (int or tuple, optional) — Stride of the convolution. (Default: 1)
- padding (int or tuple, optional) — Zero-padding added to both sides of the input (Default: 0). Here, we have selected padding = 1.
- Here, we have used a Relu() activation function for applying nonlinearity.
- ***nn.MaxPool2d(2, 2)***
- Each pooling layer i.e., ***nn.MaxPool2d(2, 2)*** halves both the height and the width of the image, so by using 2 pooling layers, the height and width are 1/4 of the original sizes.

nn.BatchNorm2d( )

Batch-normalization makes the training of convolutional neural networks more efficient, while at the same time having regularization effects.

In the sequential classification layer uses hidden layers with the sizes of 200 and 500 neurons. In the feature extraction layers, 2 max-pooling layers.

Dropout: Dropout is an effective technique to avoid overfitting [20]. Typically, dropout is applied in fully-connected neural networks applied it after the first hidden layer in the classification layer. In the **Dropout (p = 0.25)**, p = 0.25 indicates the probability at which outputs of the layer are dropped out.

C. Data

The dataset used is the Massey University Gesture Dataset 2011 containing 2515 color images cut so that the palms touch the four edges of the frame. This dataset was taken from five volunteers. Pre-processing the data is done by changing the image size which was originally an average of 500x500 pixels to 200x200 pixels and the process of adding padding so that the ratio of the image does not change.

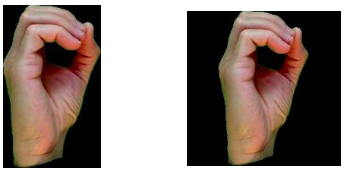
IV. RESULT AND DISCUSSION

A. Implementation

The implementation is divided into two stages, the first is the pre-processing stage and the second is the classification stage. For a complete explanation of the process, it will be explained in each sub-chapter.

1) *Preprocessing:* This stage is the process of manipulating the raw dataset before further processing at the classification stage. At this stage there are several processes, namely, Frame normalization, Estimating and generating hand keypoint datasets, and Slicing datasets.

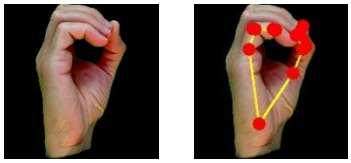
Frame Normalization: At this stage, the process of changing the image size which was originally an average size of 500x500 pixels to 200x200 pixels is carried out and the process of adding padding on each side of the image so that the ratio of the image does not change. The output of resizing and adding image padding is shown in Figure 3.



(a) (b)

Fig. 3 Frame Normalization (a) before normalization (b) after normalization

Estimating and Generating hand keypoint dataset: At this stage, the process of estimating and extracting hand keypoints is carried out on images that have been normalized at the previous stage using a pre-trained Keypoint detector and data is generated in the form of an array. The output of the process of estimating and generating hand keypoint datasets is shown in Figure 4.



(c) (d)

Figure 4 Estimating and generating hand keypoint dataset (c) result of normalization (d) result of estimating and generating hand keypoint

Slicing hand keypoint dataset: The truncation of the hand keypoint dataset data array into training data and testing data with a division of 85-15 refers to the previous research Garcia et al [6] and Kania & Markowska-Kaczmar[21] After the dataset is separated into training and testing data, the data is further processed at the Classification stage.

2) *Classification:* At the classification stage, researchers conducted experiments to find the highest accuracy value by conducting several experiments on parameters that might affect the accuracy value. So that the parameters that produce the highest accuracy value are epoch 100, batch size 3, learning rate 0.001 and using the Adam optimizer.

B. Result

The output of testing the recognition of ASL alphabet letters using CNN with a hand keypoint is in the form of probability, then the largest probability index is taken with argmax and compared with the target label, the prediction will be true if it matches the target label being tested, false if it does not match the target label. tested. The percentage of accuracy is calculated by adding the total number of correct predictions divided by the total target label and multiplied by one hundred. The formula for calculating accuracy is shown by equation 1.

$$Acc = \frac{Total\ Prediksi\ Benar}{Total\ Target\ Label} x\ 100 \quad (1)$$

Training and testing the dataset using the cross-entropy function are to get the loss value as done by previous researchers (Bheda & Radpour, 2017). Both functions are quite commonly used in image classification. The cross entropy formula is in equation 2.

Cross-entropy

$$H(p,q) = -\sum_{x} p_x \log q_x \quad (2)$$

Keterangan :

p	: label
q	: fungsi probabilitas
p_x	: index ke i
q_x	: fungsi softmax

The deepCNN Bheda & Radpour [7] method which pre-processes the data by normalizing the image size to 200x200 pixels gets higher accuracy than the Stanford deepCNN Garcia et al[6] method. Although the Stanford deepCNN method uses pre-trained GoogLeNet. However, the data used is very large and is not normalized so that the method gets lower accuracy.

Based on Table II CNN with hand keypoint in increasing accuracy American sign language gets an average accuracy of 99.55% on digits and 99.16% on alphabet. The accuracy of the digits is greater than the alphabet because the digits are easy to distinguish so they are easier to classify. Based on the result of accuracy, the proposed method has better performance than the method of previous researchers. Table 2 shows the comparison of the accuracy of the proposed method with the method of previous researchers.

TABLE II.
THE COMPARISON OF ACCURACY WITH THE PREVIOUS RESEARCHERS

Journal	Method	Accuracy
Our method	CNN + *Hand keypoint*	99.1
Kania and Markowska-Kaczmar [21]	CNN Wide Residual	93.3
Bheda & Radpour, [7]	deepCNN	82.5
Garcia et al[6]	Stanford deepCNN	72.0

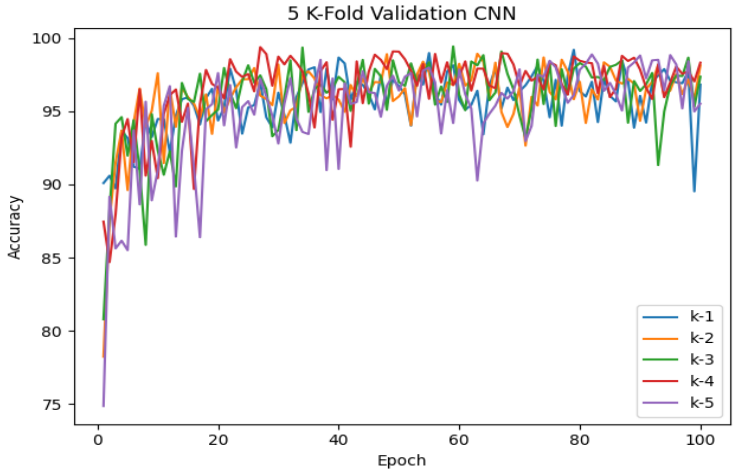


Fig. 5 Alphabet Accuracy

Figure 5 is a graph of the accuracy of the ASL alphabet as a result of testing on the a-z alphabet testing data, total 26 classes with K = 5 and 100 epochs with an average accuracy value of 99.1%

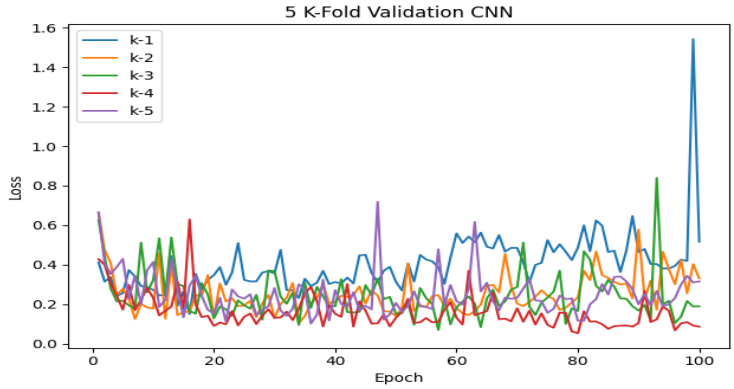


Fig. 6 Alphabet Loss Value

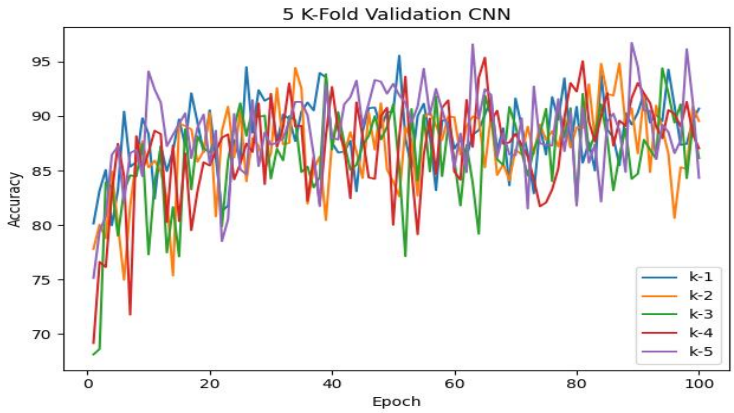


Fig. 7 Digit Accuracy

Figure 6 is an ASL alphabet loss graph as a result of testing on the a-z alphabet test data with the total of 26 classes with K=5 and 100 epochs. Figure 7 is a loss graph of the ASL alphabet as a result of testing on test data digits 0-9, with the total of 10 classes with K=5 and 100 epochs.

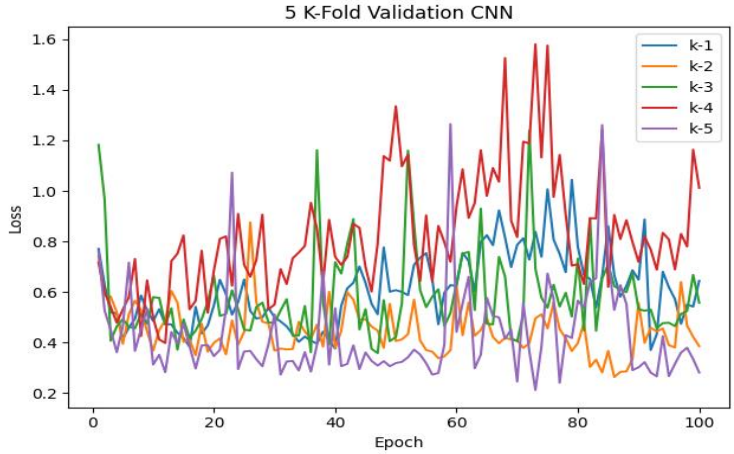


Fig. 8 Digit Loss Value

Figure 8 is a loss graph of the ASL alphabet as a result of testing on the testing data digits 0-9, with the total of 10 classes with 100 epochs. Based on the chart, the loss at the 54th epoch reaches 0.6. while the lowest and highest losses are 0.3 and 1.9 on the 10th and 9th epochs.

V. CONCLUSION

This study describes the ASL recognition method using CNN with a hand keypoint. This method uses several approaches, including using a pre-trained Keypoint detector to generate hand keypoints, pre-processing the data by normalizing the image size to 200x200 pixels and adding padding to each edge of the image. Testing the recognition of ASL using the same dataset and data sharing as the previous researchers, namely in the form of a massey dataset and dividing the dataset into 85-15.

Based on the results of CNN evaluation with hand keypoints in increasing accuracy in ASL, it successfully gets 89.5% accuracy in the alphabet. DeepCNN Bheda & Radpour[7] pre-processed the data by normalizing the image size to 200x200 pixels gets an accuracy of 82.5% higher than the Stanford deepCNN Garcia et al[6] method which got an accuracy of 72%, even though the Stanford deepCNN method uses pre-trained GoogLeNet. However, the data used is very large and is not normalized so that the method gets lower accuracy.

The results of this study still require some improvement efforts, including the pre-trained caffe model still has limited features although the use of pre-trained helps in increasing the accuracy of the classification. So, as future work, we can use better pre-trained to create and generate hand keypoint data with other model bases such as pre-trained VGG-16, ResNet50, InceptionV3 or EfficientNet or replace pre-trained completely with other models or techniques and using an improved CNN architecture for improved ASL recognition accuracy.

ACKNOWLEDGMENT

The authors would like to thank the previous researchers in particular (Garcia et al[6]., Bheda & Radpour[7], and Kania & Markowska-Kaczmar [16] because their research has inspired researchers to conduct research. this. This research is supported by the Department of Informatics, Siliwangi University.

REFERENCES

[1] Sulfayanti, Dewiani, and A. Lawi, "A real time alphabets sign language recognition system using hands tracking", *Proc. - Cybern. 2016 Int. Conf. Comput. Intell. Cybern.*, pp. 69–72, doi: 10.1109/CyberneticsCom.2016.7892569, 2017.

[2] C. M. Jin, Z. Omar, and M. H. Jaward, "A mobile application of American sign language translation via image processing algorithms", *Proc. - 2016 IEEE Reg. 10 Symp. TENSYMP 2016*, pp. 104–109, doi: 10.1109/TENCONSpring.2016.7519386, 2016.

[3] A. L. C. Barczak, N. H. Reyes, M. Abastillas, A. Piccio, and T. Susnjak, "A New 2D Static Hand Gesture Colour Image Dataset for ASL Gestures", *Res. Lett. Inf. Math. Sci*, Vol. 15, pp. 12–20, 2011.

[4] C. Vogler and D. Metaxas, "Parallel hidden Markov models for American sign language recognition", *Proc. IEEE Int. Conf. Comput. Vis.*, Vol. 1, pp. 116–122, doi: 10.1109/iccv.1999.791206, 1999.

[5] K. Y. Fok, N. Ganganath, C. T. Cheng, and C. K. Tse, "A Real-Time ASL Recognition System Using Leap Motion Sensors", *Proc. - 2015 Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discov. CyberC 2015*, pp. 411–414, doi: 10.1109/CyberC.2015.81, 2015.

[6] B. Garcia and S. A. Viesca, "Real-time Mexican Sign Language recognition", *2017 IEEE Int. Autumn Meet. Power, Electron. Comput. ROPEC 2017*, Vol. 2018-January, pp. 1–6, doi: 10.1109/ROPEC.2017.8261606, 2018.

[7] V. Bheda and N. D. Radpour, "Using Deep Convolutional Networks for gesture recognotion in american sign language", *2019 Int. Conf. Sustain. Technol. Ind. 4.0, STI 2019*, Vol. 31, No. 9, pp. 198-203 Rivera-Acosta, M., Ruiz-Varela, J. M., Orte, 2019.

[8] G. A. Rao, K. Syamala, P. V. V. Kishore, and A. S. C. S. Sastry, "Deep convolutional neural networks for sign language recognition", *2018 Conf. Signal Process. Commun. Eng. Syst. SPACES 2018*, Vol. 2018-Janua, pp. 194–197, doi: 10.1109/SPACES.2018.8316344, 2018.

[9] W. Tao, M. C. Leu, and Z. Yin, "American Sign Language alphabet recognition using Convolutional Neural Networks with multiview augmentation and inference fusion", *Eng. Appl. Artif. Intell.*, Vol. 76, July, pp. 202–213, doi: 10.1016/j.engappai.2018.09.006, 2018.

[10] Suharjito, N. Thiracitta, and H. Gunawan, "SIBI Sign Language Recognition Using Convolutional Neural Network Combined with Transfer Learning and non-trainable Parameters," *Procedia Comput. Sci.*, Vol. 179, pp. 72–80, doi: 10.1016/j.procs.2020.12.011, 2021.

[11] V. Jain, A. Jain, A. Chauhan, S.S. Kotla, A. Gautam, "American sign language recognition using support vector machine and convolutional neural network", Int. J. Inf. Technol. 13 (2021) 1193–1200, https://doi.org/10.1007/s41870-021-00617- x, 2021.

[12] M. Taskiran, M. Killioglu, and N. Kahraman, "A Real-Time System for Recognition of American Sign Language by using Deep Learning", *2018 41st Int. Conf. Telecommun. Signal Process. TSP 2018*, pp. 1–5, doi: 10.1109/TSP.2018.8441304, 2018.

[13] C. Dong, M. C. Leu, and Z. Yin, "American Sign Language alphabet recognition using Microsoft Kinect", *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, Vol. 2015-October, pp. 44–52, doi: 10.1109/CVPRW.2015.7301347, 2015.

[14] P. Rathi, R. K. Gupta, S. Agarwal, A. Shukla, and R. Tiwari, "Next Generation Computing Technologies," pp. 1–7, 2019.

[15] R. Poudel, (2018, Jul 2), GitHub, "Simple-sign-language-detector", Available: https://github.com/rrupeshh/Simple-Sign-Language-Detector, (accessed 9 Feb 2021).

[16] D. Saha, (2018, May 9). GitHub, "Sign-Language (Version1)", Available: https:// github.com/evilport2/sign-language, (accessed 9 Feb 2021).

[17] Kasapbaşi, A.; ELBUSHRA, A.E.A.; Omar, A.H.; Yilmaz, A. DeepASLR: "A CNN based human computer interface for American Sign Language recognition for hearing-impaired individuals", *Comput. Methods Programs Biomed. Update* 2022, *2*, 100048. https://doi.org/10.1016/j.cmpbup.2021.100048

[18] Alsharif, B.; Altaher, A.S.; Altaher, A.; Ilyas, M.; Alalwany, E. "Deep Learning Technology to Recognize American Sign Language Alphabet", Sensors 2023, 23, 7970. https://doi.org/10.3390/s23187970, 2023

[19] T. Simon, H. Joo, I. Matthews, and Y. Sheikh, "Hand keypoint detection in single images using multiview bootstrapping", *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 4645–4653, doi: 10.1109/CVPR.2017.494, 2017.

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting Nitish", *J. Mach. Learn. Res. 15*, Vol. 299, No. 15, pp. 1929–1958, doi: 10.1016/0370-2693(93)90272-J, 2014.

[21] Kania, K, Markowska-Kaczmar U (2018), "American sign language fingerspelling recognition using wide residual networks", In: International conference on artificial intelligence and soft computing, Springer, pp 97–107, 2018.