Computer Engineering and Applications Vol. 10, No. 2, June 2021

Classification of Finger Spelling American Sign Language Using Convolutional Neural Network

Anna Dwi Marjusalinah[1], Samsuryadi Samsuryadi[2,*], Muhammad Ali Buchari[3]

[1]*Graduate Program in Computer Science, Faculty of Computer Science, Universitas Sriwijaya,*
[2]*Department of Computer Science Faculty of Computer Science, Universitas Sriwijaya*
[3]*Department of Computer System, Faculty of Computer Science, Universitas Sriwijaya*
**samsuryadi@unsri.ac.id*

ABSTRACT

Sign language is a combination of complex hand movements, body postures, and facial expressions. However, only a limited number of people can understand and use it. A computer aid sign language recognition with finger spelling style utilizing a convolutional neural network (CNN) is proposed to reduce the burden. We compared two CNN architectures such as Resnet 50, and DenseNet 121 to classify the American sign language dataset. Several data splitting proportions were also tested. From the experimental result, it is shown that the Resnet 50 architecture with 80:20 data splitting for training and testing indicates the best performance with an accuracy of 0.999913, sensitivity 0.998966, precision 0.998958, specificity 0.999955, F1-score 0.999913, and error 0.0000898.

Keywords: American Sign Language, Finger Spelling, Resnet 50, DenseNet 121.

1. INTRODUCTION

Sign language is a combination of complex hand movements, body postures, and facial expressions. The three combinations are expected to convey meaning to people with hearing and speech impairments and between persons with disabilities and normal people [1]. One of the most widely used sign languages is American-style sign language, better known as ASL (American Sign Language). This language style is used more standard than other language styles because it is easier and does not have many variations or dialects in spelling [2].

There is a limited number of people who can understand and use sign language as a skill. Furthermore, not everyone can use it. Only a small number of groups who need to use sign language can understand it [3]. As a result, this gap creates a lot of social isolation between them in their daily life. To bridge this, we can use technology by applying artificial intelligence so that computers can learn to recognize ASL movements and then translate them into everyday language.

Hand gesture recognition using computer-human interaction techniques is currently prevalent [4], [5]. Computer vision based in hand gesture recognition is a technique that is explored with various implementations using conventional image processing by applying techniques such as skin detection, hand segmentation, and hand motion tracking [3].

Hand gesture recognition is undergoing a fairly slow research transformation as it poses many challenges to the machine learning process. Many factors influence an image to be good data [6]. Among them are lighting conditions, shooting backgrounds, use of hands, shooting directions and skin tones which vary widely by

ethnicity. Some of these factors must be considered so that the processed data can produce a good outcome.

The CNN method is still currently recognized as the best method of image recognition [7]. CNN can be a solution for recognizing and classifying hand movements in the form of images because CNN has a significant conceptual framework such as weight distribution, local perception area, and down sampling space. In this method, the displacement, distortion, and scaling characteristics are relatively unchanged [8]. However, in previous research, it is not known that CNN architecture (such as Lenet, Alexnet, VGG, Resnet, etc.) which can produce the best performance in recognition and classification of images in ASL sign language. Therefore, this study will focus on the classification of ASL sign language with the CNN method using several architectures to obtain robust features and improve accuracy in ASL classification by performing parameter tuning.

2. RELATED WORKS

Sign Language as an alternative means of communication for individuals with hearing and speech impairments is very important. Research on the development of sign language recognition using artificial intelligence technology has been started since 1998 by S. Naidoo using traditional machine learning methods, namely the support vector machine (SVM) to classify South African Sign Language (SASL) [9]. The introduction of Sign Language in general involves several processes, namely segmentation, feature extraction and classification,

The main objective of the segmentation process is to remove background and noise, leaving only a Region of Interest (ROI). Furthermore, in the feature extraction process, the ROI feature file will be extracted. This feature can be in the form of shape, color, background and others. In the context of Sign Language recognition, these features are basically analogous to the identity of each sign language sign. Furthermore, the extracted features will undergo a classification where the features of each move will be grouped and this will be used as a database to match the new sign language and will be classified into each group.

The earliest research on classifiers was started by Pegeault and Bowden (2011), both of which used a Gabor filter to extract features which were then used to train multi-class random forests to develop classifiers for 24 ASL letters [8]. Subsequent research uses various types of combination methods of feature extraction with the Deep Belief Network (DBN) method for classification [10]. Research on sign language classifiers continues to grow, including using the Machine Learning method [11]–[13]. In the research of Bhattacharya et al., (2019), from several tested models, the best accuracy results were obtained using the SVM algorithm and the SURF extraction feature of 91.35%. However, in this study the data used was still small and the resulting level of accuracy could still be improved.

Recent studies using deep learning (DL) methods [1], [3]. The use of DL is considered preferable because it does not require minimal feature extraction and pre-processing. This means more time-efficiency increases and better results as computers learn more to manage images. Research with the DL method which is currently popular for image processing is the convolutional neural network (CNN) method [8], [9], [14], [15].

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

3. MATERIALS AND METHODS

3.1. DATASET

In the data preparation stage, 72,000 ASL images were divided into several data ratio. We use 90:10, 80: 20, 70: 30 and 60:40 for training data and testing data ratio. The dataset is sourced from https://www.kaggle.com/grassknoted/asl-alphabet?select=asl_alphabet_train. This dataset measures 200 x 200 pixels in the form of RGB (Red, Green, Blue) images. The image pixel range in this dataset is 0-255 for each channel.

3.2. PRE-PROCESSING DATA

There are three stages in pre-processing the first data is the resizing process, resizing is the process of changing the image size so that it can be processed on a predetermined architecture. The second is the process of the scaling feature, the scaling feature is the process of changing the pixel range from 0-255 to 0-1, so that the data initialization process is faster and optimal. Third is the One Hot Encoding process. One hot encoding process is carried out on label testing and training.

3.3. CNN MODEL

The ASL finger spelling classification stage requires a model that is generated from the training results. Before conducting training, the architecture of CNN must be defined first. The architecture developed in this research consists of several architectures, including Resnet50 and DenseNet 121. In [16] research, the ResNet50 architecture consists of several layers, including input layer, convolution layer, pooling layer, deconvolution and output layer. The input layer has an output shape of 256x256x1. Whereas the convolutional layer has a different kernel size, feature map and output shape, but has the same stride, which is 2 except for the convolutional layer 5. The pooling layer and deconvolution separate each convolutional layer. For clarity, Table 1 illustrates the ResNet50 architecture.

Further research [17] introduced the Dense Convolutional Network (DenseNet). DenseNets has several interesting advantages, namely that it can reduce vanishing-gradient problems, strengthen feature propagation, encourage feature reuse, and can substantially reduce the number of parameters [18]. The DenseNet architecture consists of several layers, including the convolution layer, pooling, the Dense block and the Transition layer. Table 2 provides a summary of the DenseNet architecture.

TABLE 1.
ResNet 50 Architecture

Layer name	Number of Filter and Filter Size	Output Size
First Convolution Blok	7x7, 64 Stride 2	112 x 112
Max Pooling	3 x 3, Stride 2	
Second Convolution Blok	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	56 x 56
Third Convolution Blok	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	28 x 28
Forth Convolution Blok	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	14 x 14
Fifth Convolution Blok	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	7 x 7
Average pool layer	[1 x 1]	1 x 1
Fully Connected layer	1000	
Softmax	*SoftMax*	
Total Parameter	25,996,184	

TABLE 2.
DenseNet Architecture

Layer	Output Size	DenseNet-201 (k=32)
Convolution layer	112x112	7x7 conv, stride 2
Pooling layer	56x56	3x3 max pool, stride 2
Block Dense (1)	56x56	$\begin{bmatrix} 1x1 \text{ conv} \\ 3x3 \text{ conv} \end{bmatrix}$ x 6
Transition layer (1)	56x56	1x1 conv
	28x28	2x2 avg pool, stride 2
Block Dense (2)	28x28	$\begin{bmatrix} 1x1 \text{ conv} \\ 3x3 \text{ conv} \end{bmatrix}$ x 12
Transition layer (2)	28x28	1x1 conv
	14x14	2x2 avg pool, stride 2
Block Dense (3)	14x14	$\begin{bmatrix} 1x1 \text{ conv} \\ 3x3 \text{ conv} \end{bmatrix}$ x 48
Transition layer (3)	14x14	1x1 conv
	7x7	2x2 avg pool, stride 2
Block Dense (4)	7x7	$\begin{bmatrix} 1x1 \text{ conv} \\ 3x3 \text{ conv} \end{bmatrix}$ x 32
Classification layer	1x1	7x7 global average pool
		1000D Fully Connected, Softmax

Furthermore, the type of pooling operation used is max pooling with a kernel width of 2x2. This pooling operation is only used at the last convolution layer. In the CNN classification stage, there are two stages, namely training and testing. In general, the process flow in the CNN classification with various architectures is shown in Figure 1.

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

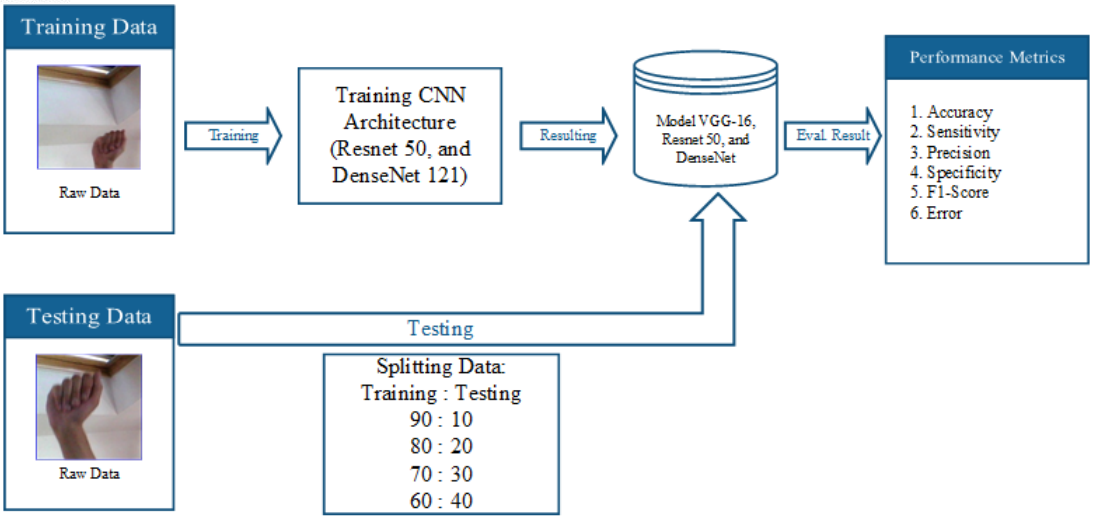

FIGURE 1. Flow Diagram CNN for Training and Validating

3.4. MEASUREMENT METRICS PERFORMANCE

In this study, the accuracy of the classification process will be calculated. The parameters for testing the classification process are the recall value / sensitivity (R), precision (P), Specificity (S), F1 score (F1), Accuracy (A), and Error (E). The equation for calculating these parameters is:

$$\boldsymbol{P} = \frac{TP}{TP + FP} \tag{1}$$

$$\boldsymbol{R} = \frac{TP}{TP + FN} \tag{2}$$

$$\boldsymbol{S} = \frac{TN}{TN + FP} \tag{3}$$

$$\boldsymbol{F_1} = 2\,x\frac{P\,x\,R}{P + R} \tag{4}$$

$$\boldsymbol{A} = \frac{TP + TN}{TP + \ TN + FP + FN} \tag{5}$$

$$\boldsymbol{E} = 1 - A \tag{6}$$

With TP, FP, and FN are true positive, false positive, and false negative values generated from the configuration matrix as shown in Table 3.

TABLE 3.
Confussion Metrics

		Predicted Value	
		Positive	Negative
True Label	Positive	TP	FN
	Negative	FP	TN

4. RESULTS AND DISCUSSION

In this study, testing was carried out 4 times by dividing the dataset into 90:10, 80:20, 70:30 and 60:40 for training data and validation data. Each dataset was tested using 2 models, the first model used the ResNet 50 architecture, and the second model used the DenseNet 121 architecture. The parameters used in this study included a batch size of 32, epoch 100 and using the ADAM optimizer method.

4.1. EXPERIMENTAL RESULT OF RESNET 50

To find out how well the ResNet 50 architecture performs, tests are carried out using several parameters. The performance to be seen includes sensitivity, precision, specificity, f1-score, error, and accuracy. Table 4 shows the performance measurement results on the ResNet 50 architecture in classifying ASL finger spelling.

The performance measurement of the ResNet 50 architecture shows excellent results. It can be seen in Table 2 that the performance results for each parameter (except for errors) reach a value above 0.95 (95%). Meanwhile, the error value detected in the ASL finger spelling classification case is also very small.

TABLE 4.
ResNet 50 Result

Dataset Split	sensitivity	precision	specificity	f1-score	error	accuracy
90:10	0.992038	0.991667	0.999638	0.999306	0.000694	0.999306
80:20	0.998966	0.998958	0.999955	0.999913	0.0000868	0.999913
70:30	0.963446	0.958056	0.998186	0.996505	0.003495	0.996505
60:40	0.964693	0.957292	0.998151	0.996441	0.003559	0.996441

The accuracy obtained in the ASL finger spelling classification using the ResNet 50 architecture in all dataset testing processes obtained excellent results. In 4 stages of testing with the distribution of different datasets, the accuracy value is above 0.99 (99%). Even in the dataset with a test size of 0.2, the accuracy is 0.9999 (99.99%). This means that the ResNet 50 architecture can properly classify the entire class in ASL finger spelling cases.

The best performance measurement results in the ASL finger spelling classification model using ResNet 50 architecture are found in the dataset with a test size of 0.2 with a sensitivity value of 0.998966, precision 0.998958, specificity 0.999955, f1-score 0.999913, error 0.0000898 and accuracy 0.999913.

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

4.2. EXPERIMENTAL RESULT OF DENSENET 121

Performance measurement on the ASL finger spelling classification model using the DenseNet 121 architecture can be seen in Table 5. The results of measuring the performance of the DenseNet 121 architecture for ASL finger spelling classification cases can be said to be very good. In each test with a different test size, the performance measurement results for all parameters (except error) are more than 0.93 (93%) and the error value is small.

The accuracy value obtained in measuring the performance of the DenseNet architecture reaches a value above 0.98 (98%). This means that the DenseNet 121 architecture can also perform ASL finger spelling classification tasks well. There are very few errors in classifying ASL finger spelling.

The best results for measuring the performance of ASL finger spelling classification using DenseNet 121 architecture are found in the dataset with a test size of 0.2 with a sensitivity value of 0.987116, precision 0.986458, specificity 0.999412, f1-score 0.998872, error 0.001128 and accuracy 0.998872.

TABLE 5.
DenseNet 121 Result

Dataset Split	sensitivity	precision	specificity	f1-score	error	accuracy
90:10	0.937809	0.85	0.993627	0.9875	0.0125	0.9875
80:20	0.987116	0.986458	0.999412	0.998872	0.001128	0.998872
70:30	0.98183	0.979583	0.999119	0.998299	0.001701	0.998299
60:40	0.980734	0.979896	0.999127	0.998325	0.001675	0.998325

4.3. COMPARISON OF RESNET 50 AND DENSENET 121

Based on Table 4 and Table 5, it can be compared the performance measurement results of the two ASL finger spelling classification models using the ResNet 50 architecture and the DenseNet 121 architecture. The best results from each architecture are compared to obtain the best value between the two models that have been tested.

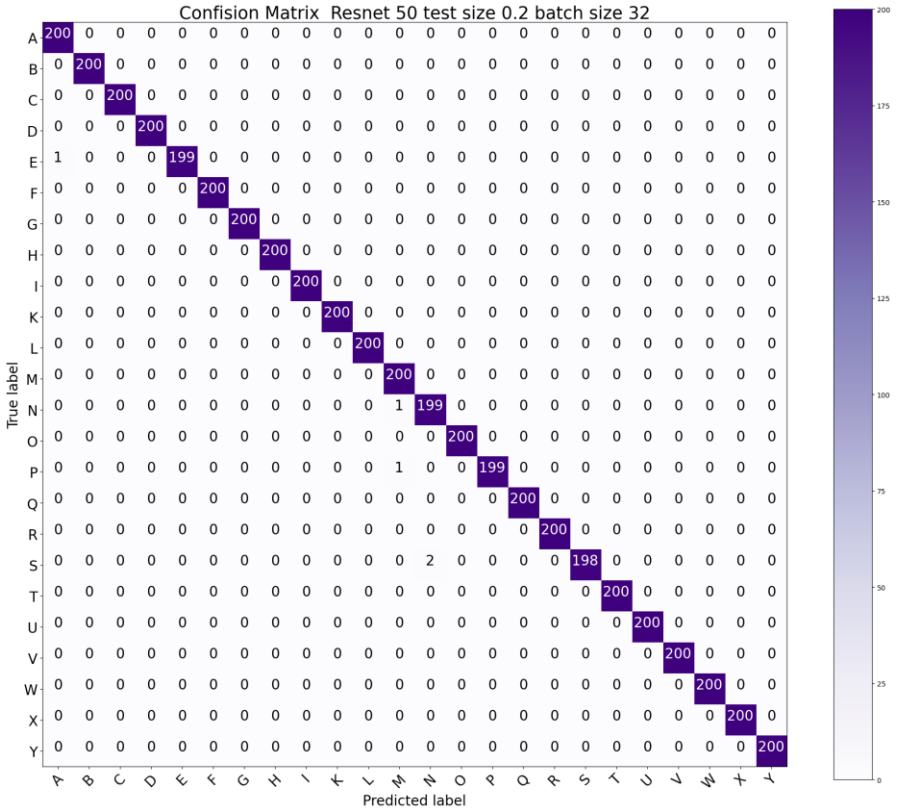

FIGURE 2. Confusion Metric ResNet 50 with test size 0.2 (20%)

In the ResNet 50 architecture, the best performance measurement results are obtained at test size 0.2 with an accuracy value of 0.999913 (99.99%). Whereas on the DenseNet 121 architecture the best performance measurement results were obtained at the 0.2 test size with an accuracy value of 0.998872 (99.88%). Each architecture gets the best results at test size 0.2, where the dataset used is a set with a ratio of 80:20. This means that a model can produce good performance if the proportion of the training dataset is more than the validation data. However, we must also look at the distribution value of the dataset so that a robust model is obtained. In the case of ASL finger spelling classification, the best data comparison value is 80:20.

From each value obtained, it can be concluded that the best ASL finger spelling classification model is the model using ResNet 50 architecture at test size 0.2. The results of measuring the ResNet 50 architecture in the size 2 test are obtained from the value of the configuration metric shown in Figure 2. Furthermore, the values contained in the configuration metric are translated into the performance measurement parameters shown in Table 6. It shows the results of performance measurements in each class. ASL finger spelling classification. Then to get the overall result, it is obtained from the average value of the entire class. So that the performance measurement results obtained on the ResNet 50 test size 0.2 architecture, namely sensitivity 0.998966, precision 0.998958, specificity 0.999955, f1-score 0.999913, error 0.0000898 and accuracy 0.999913.

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

TABLE 6.
Best Accuracy of ResNet 50

	class	sensitivity	precision	specificity	f1-score	error	accuracy
0	A	0.995025	1	1	0.997506	0.000208333	0.999792
1	B	1	1	1	1	0	1
2	C	1	1	1	1	0	1
3	D	1	1	1	1	0	1
4	E	1	0.995	0.999783	0.997494	0.000208333	0.999792
5	F	1	1	1	1	0	1
6	G	1	1	1	1	0	1
7	H	1	1	1	1	0	1
8	I	1	1	1	1	0	1
9	K	1	1	1	1	0	1
10	L	1	1	1	1	0	1
11	M	0.990099	1	1	0.995025	0.000416667	0.999583
12	N	0.99005	0.995	0.999783	0.992519	0.000625	0.999375
13	O	1	1	1	1	0	1
14	P	1	0.995	0.999783	0.997494	0.000208333	0.999792
15	Q	1	1	1	1	0	1
16	R	1	1	1	1	0	1
17	S	1	0.99	0.999565	0.994975	0.000416667	0.999583
18	T	1	1	1	1	0	1
19	U	1	1	1	1	0	1
20	V	1	1	1	1	0	1
21	W	1	1	1	1	0	1
22	X	1	1	1	1	0	1
23	Y	1	1	1	1	0	1
24	average	0.998966	0.998958	0.999955	0.999913	8.68056E-05	0.999913

5. CONCLUSION

This study explores the American sign language classification process using a convolutional neural network (CNN) algorithm. The CNN architecture used is Resnet 50 and DenseNet 121. In addition, the distribution of the dataset was also tested in this study. To measure the performance of each model, several measurement metrics are used. From the experimental results, the Resnet 50 architecture with 80:20 data splitting for training and testing respectively shows the best performance with an accuracy of 0.999913, sensitivity 0.998966, precision 0.998958, specificity 0.999955, f1-score 0.999913, and error 0.0000898.

ACKNOWLEDGEMENT

This work was supported by Database and Big Data Laboratory. We thank to Mrs. Dinda Lestarini as the head of Database and Big Data Laboratory, Faculty of Computer Science, Universitas Sriwijaya.

REFERENCES

[1] H. B. D. Nguyen and H. N. Do, "Deep learning for American sign language fingerspelling recognition system," *2019 26th Int. Conf. Telecommun. ICT 2019*, pp. 314–318, 2019, doi: 10.1109/ICT.2019.8798856.

[2] B. Vivek and N. Dianna Radpour, "Using Deep Convolutional Networks for Gesture Recognition in American Sign Laguage," 2017.

[3] L. Kumar, "Real-Time Finger Spelling American Sign Language Recognition Using Deep Convolutional Neural Networks," 2018.

[4] K. Bantupalli, "American Sign Language Recognition Using Machine Learning and Computer Vision," 2018.

[5] D. Kelly, "Computational Models for the Automatic Learning and Recognition of Irish Sign Language by," *Thesis*, p. 293, 2010.

[6] K. Sahithya, P. Road, K. Sahithya, and P. Road, "Sign Language Translator Using Machine Learning," vol. 13, no. 4, pp. 1–5, 2018.

[7] K. Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, "Convolutional neural networks: an overview and application in radiology. Insights into Imaging," 2018.

[8] A. Salem and S. Vadera, "A Convolutional Neural Network to Classify American Sign Language Fingerspelling from Depth and Colour Images," 2017.

[9] R. Daroya, D. Peralta, and P. Naval, "Alphabet Sign Language Image Classification Using Deep Learning," *IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON*, vol. 2018-Octob, no. October, pp. 646–650, 2018, doi: 10.1109/TENCON.2018.8650241.

[10] L. Rioux-Maldague and P. Giguere, "Sign language fingerspelling classification from depth and color images using a deep belief network," *Proc. - Conf. Comput. Robot Vision, CRV 2014*, pp. 92–97, 2014, doi: 10.1109/CRV.2014.20.

[11] 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, 2016, doi: 10.1109/TENCONSpring.2016.7519386.

[12] A. Bhattacharya, V. Zope, K. Kumbhar, P. Borwankar, and A. Mendes, "Classification of Sign Language Gestures using Machine Learning," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 8, no. 12, pp. 97–103, 2019, doi: 10.17148/IJARCCE.2019.81219.

[13] R. Kurdyumov, P. Ho, and J. Ng, "Sign Language Classification Using Webcam Images," pp. 1–4, 2011, [Online]. Available: http://cs229.stanford.edu/proj2011/KurdyumovHoNg-SignLanguageClassificationUsingWebcamImages.pdf.

[14] Z. Parcheta and C. D. Martinez Hinarejos, "Sign Language Gesture Classification using Neural Networks," no. November, pp. 127–131, 2018, doi: 10.21437/iberspeech.2018-27.

[15] R. Gupta and S. Rajan, "Comparative Analysis of Convolution Neural Network Models for Continuous Indian Sign Language Classification," *Procedia Comput. Sci.*, vol. 171, no. 2019, pp. 1542–1550, 2020, doi: 10.1016/j.procs.2020.04.165.

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

[16] Q. Ji, J. Huang, W. He, and Y. Sun, "Optimized deep convolutional neural networks for identification of macular diseases from optical coherence tomography images," *Algorithms*, vol. 12, no. 3, pp. 1–12, 2019, doi: 10.3390/a12030051.

[17] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 2261–2269, 2017, doi: 10.1109/CVPR.2017.243.