2022 IEEE Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI) | 978-1-6654-7719-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/IATMSI56455.2022.10119336

Effective Recognition System of American Sign Language Alphabets using Machine Learning Classifiers, ANN and CNN

Diponkor Bala
Dept. of Computer Science and Engg.
Islamic University
Kushtia, Bangladesh
diponkor.b@gmail.com

Mohammad Alamgir Hossain
Dept. of Computer Science and Engg.
Islamic University
Kushtia, Bangladesh
alamgir@cse.iu.ac.bd

Mohammad Anwarul Islam
Department of Mathematical Sciences
Georgia Southern University
Georgia, USA
mislam@georgiasouthern.edu

Mohammed Mynuddin
Dept. of Electrical and Computer Engg.
Georgia Southern University
Georgia, USA
mm40771@georgiasouthern.edu

Md. Shamim Hossain
School of Computer Science and Tech.
Uni. of Science and Technology of China
Hefei, China
shamim2@mail.ustc.edu.cn

Md. Ibrahim Abdullah
Dept. of Computer Science and Engg.
Islamic University
Kushtia, Bangladesh
ibrahim@cse.iu.ac.bd

***Abstract*—People who cannot talk are called audibly impaired, and they communicate with others through other means. The most popular method of communication is through sign language. American Sign Language (ASL) is the de-facto standard for sign languages taught globally. Automated sign language recognition tries to bridge the gap. Convolutional neural networks are the method of choice these days when it comes to the classification of multiclass images. To recognize ASL alphabets, we used a CNN, traditional machine learning classifiers, and an artificial neural network. The Sign Language MNIST dataset has a total of 34,627 image data, of which 27,455 and 7172 are training and test data respectively. Except for J and Z, the dataset comprises 24 alphabets. We used the training dataset to train our CNN, ANN, and other machine learning models. Then examined our proposed CNN model as well as other models including ANN on the test dataset to check how well they recognize ASL alphabets correctly. The traditional classifiers such as Linear Regression, Logistic Regression, Random Forest, SVM, and ANN were able to achieve an accuracy of 71.94%, 90.16%, 98.63%, 97.92%, 82.96% respectively whereas the proposed CNN model achieved 100% of accuracy on the unseen test data.**

***Index Terms*—American Sign Language, ASL Alphabet, Recognition, Machine Learning Classifier, ANN, CNN;**

I. INTRODUCTION

Those who have trouble hearing or speaking can communicate with others through hand and facial gestures. When communicating with the deaf, sign language should be used. Sign language is vital for deaf and hearing-impaired individuals to communicate. Most individuals don't fully grasp sign language. As a result, there is a significant lack of communication between regular individuals and disabled people [1]. As a result, several technological breakthroughs have been made to assist the deaf. Another option is to employ technology to convert sign language into a language that the broader population understands. Clearly, this strategy will help handicapped and non-impaired people communicate. Worldwide, over a hundred sign languages are used by disabled individuals including Indian, American, and Russian. They employ sign language, which is used by around a million Indians. United Nations statistics place American Sign Language (ASL) at number four among the continent's official languages. More than 30 countries with English as the official language utilize ASL, including England and Wales [2]. Americans and others throughout the world use ASL for communication.

Sign language allows deaf people to express themselves using complicated hand, finger, and facial gestures. ASL is widely regarded as a trustworthy mode of communication. It has as many dialects as other languages, like Italian or German. Many people who are deaf or hard of hearing can benefit greatly from sign languages. People with disabilities have more joy, hope, and power when they can communicate with others in ASL. ASL is a set of 26 gestures that represent words from the English language dictionary. People devised 19 alternative hand forms to transmit 26 manual alphabets in American Sign Language. Due to the limited variety of hand signs now available, we can communicate in multiple alphabets by changing the hand shape's orientation [2]. For instance, "K" and "P" are frequently used alphabets. The numerals 0 through 9 can also be used by hand gestures. Even though there aren't any hand gestures in English that represent words or concepts, there are many facial and hand gestures that can be used to convey meaning.

However, the current research tries to explore the performance of different types of ML models to classify ASL efficiently and correctly. Because of new technology, computer vision is becoming more and more popular as a way to help people with their daily lives. When it comes to categorizing images, convolutional neural networks (also known as CNNs) have proven to be quite effective [3], [4]. In this study, we applied computer vision technology with CNN architecture to recognize the ASL Alphabet by using different ML models to classify the alphabets of sign language. We aimed to develop an efficient recognition system as a contribution of this study, which can recognize ASL alphabets efficiently with the highest accuracy. The extracted features from the image data were used as inputs for the proposed CNN, ANN, and some other machine learning models. We compared the performance of our proposed model to that of other models.

Here's how the rest of the text is laid out: Section II discusses relevant literature, while Section III provides a concise overview of the data set. In Section IV, we give a high-level description of the suggested method. In Section V, we give an overview of the evaluation metrics we used to measure how well the proposed model worked. In Section VI, we talk about the study's findings and what they mean, and in Section VII, we say what the study's conclusions are.

II. Related Works

Researchers have tried many methods to recognize sign languages from numerical and pictorial data during the last few decades. Several previous works have been completed using machine learning classification algorithms. Deep learning models and SVM have been used for early gesture classification in static ASL. The following is a description of some existing work:

A. S. Konwar *et al.* [5] showed the HSV color model and an edge detection method as part of a system for finding ASL from hand gesture images. They have obtained a 65% ASL recognition rate. S. Upendran *et al.* [6] proposed a scheme to recognize and interpret ASL alphabets. They have extracted features from images using a technique called PCA, and they have classified hand movements using a KNN classifier. The proposed system accuracy rate is 77.29%. Bheda *et al.* suggested an approach to interpreting ASL numerals and alphabets that was implemented in the architecture of a deep convolutional neural network. They have achieved 82.5% and 97% accuracy on alphabet gestures and digits, respectively. Srinath S *et al.* [7] presented a segmentation technique-based classification approach for the recognition of sign language. The model that was suggested is 86.67% accurate when it comes to recognizing ASL alphabet gestures.

J. R. Pansare *et al.* [8] suggested an edge orientation histogram technique-based hand gesture recognition system for ASL alphabet recognition. They correctly identified 88.26% of the ASL alphabets. Singha *et al.* [9] was created an image dataset comprising 240 images with 24 indigenous signs in the ISL language and published in Science. The Eigen values were then recovered and classified based on their Euclidean distance from the images. This method yielded a 97% of success rate. G. A. Rao *et al.* [10] designed a CNN framework for sign language identification and achieved a 92.88% recognition rate. Using the Transfer Learning method, Rathi *et al.* [11]suggested making a system that could recognize the alphabets of ASL. They used the MobileNet and InceptionV3 models and achieved 95.06% and 93.36% respectively.

Following our reading of the publications listed above, we attempted to evaluate several models and enhance the precision of ASL identification by utilizing machine learning classifiers, artificial neural networks (ANN), and convolutional neural networks (CNN).

III. Dataset Description

One of the keys and critical components of every investigation is data collection. Acquiring data is the basis of any research and should be carefully collected. We have chosen a dataset named "Sign Language MNIST" for our research. Sign Language MNIST [12] is a dataset used by researchers who want to build a model for sign language alphabet classification. It's from the MNIST database, which is a massive library of image-based handwritten numbers used to recognise ASl alphabets. MNIST sign language comprises 28x28 pixels (784 total) per sample. MNIST pictures in sign language are all colored instead of black and white. Hence, the letters J and Z need motion, so they are not included. The dataset contains 34627 images, 27455 training images, and 7172 testing images. This constitutes approximately one-half of the total MNIST dataset. Each image has a label (0–8, 10–24). There is no 9 and no 25, for we exclude the letter J and letter Z. All images in the dataset are saved in a csv file. Each row represents a single image. The first column is label (the class number for this image). From the second column to the 785th column, there are 784 pixels. Each pixel has grayscale values between 0-255, for which the image is converted to gray. These images show multiple people repeating the gesture against different backgrounds.

IV. Methodology

The proposed system consists of four main stages, which include dataset preprocessing, network creation, model training and testing, and identification of gestures by means of a CNN model that has been trained. Fig. 1 is a flowchart that shows how the proposed system for recognizing ASL would work.

A. Data Preprocessing:

Feature Scaling: To start, we prepared the dataset so that various algorithms could accurately predict the sign language alphabet labels. There is already a dataset that can be used for both training and testing. When we analyzed our dataset, there was a balance. In addition to this, we generate a randomly selected image for the purpose of predicted class verification and preserve the labels within each image in a database. To minimize the visual impact of the illumination differences, we utilized grayscale normalization [13]. Also, when given data between [0, 1], the CNN converges more quickly than when given data between [0, 255]. Following the completion of the normalization step, we then split the training set into two distinct parts: training and validation. We now have three different types of data: test data, validation data, and training

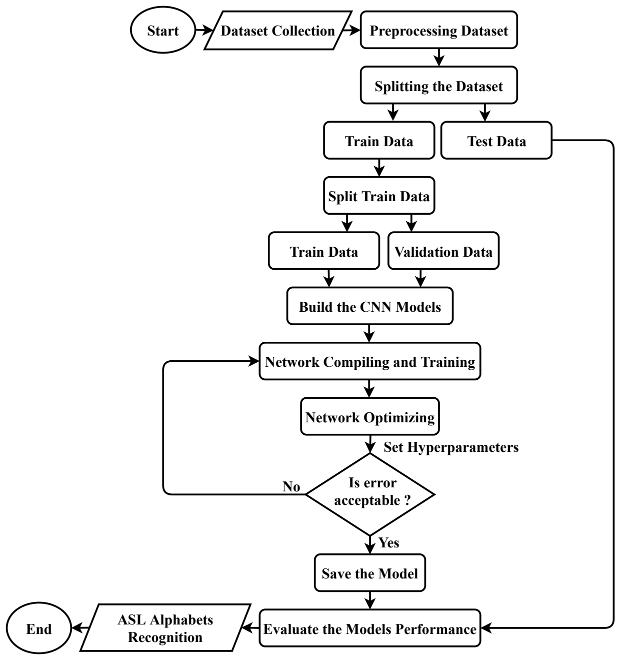

Fig. 1. Flowchart of the Proposed Methodology

data. In this case, each of the 27455 images that were entered has 784 columns. In order to use this information with the CNN model, we converted it from a representation with a single dimension to one with three dimensions. So we made each row of 784 columns into a (28,28,1) matrix. If the normalized value is represented by x' then,

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{1}$$

to which x refers to the image's initial intensity.

Data Augmentation: The term "augmentation" refers to expanding the size of the training dataset. Data augmentation creates additional images from a single image. This could make our dataset more diverse, and it is also used to increase the number of training instances to make it less likely that the model is overfitted [14]. The Keras image preprocessing package was used to achieve this purpose. Our data augmentation includes:

- Randomly rotated by 10 degrees
- Randomly shifted in width by 10%
- Randomly shifted in height by 10%
- Randomly zooming
- Randomly flip inputs horizontally

B. Model Creation:

Machine Learning Classifier: For the classification and recognition of ASL, we have used Linear Regression, Logistic Regression, Random Forest, and SVM classifiers.

ANN Model: Our model is built using the open-source Keras library. So we imported Keras and all its dependencies. So we started with a sequential model. Our network's input layer requires 784 rows of data. The input dimension is 784, which is the dataset's column count. We employed four hidden layers, each with 404 nodes, and the Rectified Linear Unit (ReLU) Activation Function [15] for their speed. We want our model to be dense at each layer. To avoid overfitting, we inserted a dropout between the concealed layers. We've also introduced batch normalization between layers to speed up our model's performance. Normalization adds layers to our neural network that normalize information from the input layer. There are a total of 820,549 trainable parameters in the model.

The output layer has 25 nodes and uses softmax activation [16]. We employed sparse categorical cross-entropy to hit our target. Lastly, we make a compiled version of our model and use the "Adam" optimizer (stochastic Gradient Descent Method) to find the minimum value of the cost function [17]. To optimize with the "Adam" algorithm, we set $\beta 1$ = 0.9 and $\beta 2$ = 0.999. We've also used "ReduceLROnPlateau" to slow down the learning rate so our model improves and this was applied during the model training on the training dataset.

Proposed CNN model: Specifically designed for data that can be represented as grids, the Convolutional Neural Network is an advanced form of neural network. Given that this input can be seen as a 2D grid of pixels, CNN is frequently employed for image recognition. At least one layer of CNNs uses the convolution technique. Because of their distinct qualities, CNNs can deal with more data inputs than most systems.We presented a CNN architecture of eight convolutional layers, three pooling layers, two dropout layers, and a fully connected layer for ASL alphabet recognition. Fig. 2 presents the CNN model architecture that has been proposed. Below is an illustration of the suggested model for CNN, which can be found in its description:

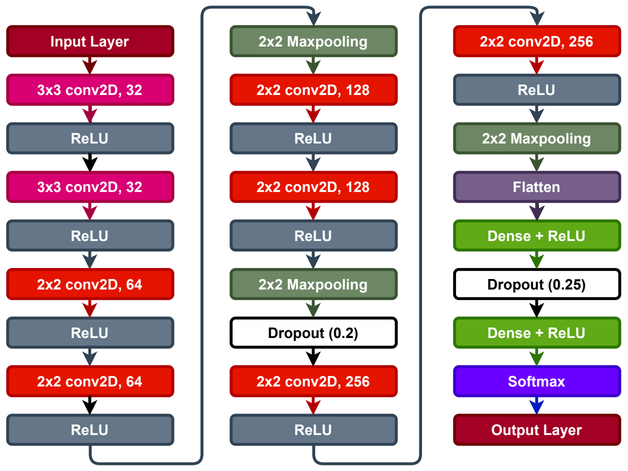

Fig. 2. Proposed CNN Model Architecture

- Both the hidden and output levels can communicate with the input layer. Throughout this investigation, the image form of the input layer is (28, 28, 1).
- Our CNN model uses a mathematical approach called convolution. Convolution is the process of figuring out what features are in an image and putting them into a

feature vector [16]. For the input image (X) and kernel k, the 2D convolutional operator is defined as:

$$(X * K)(i,j) = \sum_m \sum_n K(m,n) X(i-m, j-n) \quad (2)$$

where $*$ represents mathematical representation of convolution operation, the k matrix moves over the input data matrix with stride parameter.

- To ensure that nonlinearity is maintained in our proposed model, We used the activation function of the rectified linear unit (ReLU) in this study [15]. The equation for the ReLU function can be written as follows:

$$ReLU(x) = \begin{cases} max(0,x) & \text{for x>0} \\ 0 & \text{for x} \leq 0 \end{cases} \quad (3)$$

where x represents the value that is being received by the neuron.
- Pooling is another common procedure in Convolutional Neural Networks. The pooling layer's output is a condensed version of the input. More valuable information is presented by the pooling operation. A comparison of different pooling approaches revealed that Max Pooling outperforms average pooling and attention pooling [16].
- It is necessary to use the dropout layer after the convolution layer in order to prevent overftting during the processing process [18]. In the proposed CNN, we implemented two dropout layers with dropout rates of 20% and 25%, respectively, in each layer.
- The flatten layer turns images into a one-dimensional array, allowing each pixel to be connected to the next layer. It combines all the features retrieved from the layers. Then we add and transmit data through layers [16].
- The dense layers function is responsible for determining the relationship between all of the characteristics that have been provided to it without the use of any more input parameters than convoluted layers [16].
- The Softmax activation function was employed to create the suggested CNN architecture. When using the softmax classifier, linear input data will be converted into probabilistic attributes and characteristics of all 25 different kinds of classes, and then the probability of occurrence will be compared to the actual outcome [16]. A definition of the softmax function would look something like this:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} fori = 0, 1, \ldots, K \quad (4)$$

This function takes a K-dimensional input vector z and returns a K-dimensional vector of values within the range 0 to 1 that sum to 1.

We employed categorical cross-entropy to label targets. Finally, we construct our model and apply the stochastic Gradient Descent method-based "Adam" optimizer to minimize the cost function [17]. The learning rate was set to 0.001, the batch size was set to 250, and the epochs were set to 100. During the model training stage, we used a callback to see if our validation set accuracy increased after 2 (patience level) epochs. Then we will reduce our learning rate by half. This was followed by a training dataset.

C. Model Training and Testing:

Our model was trained with the help of the training dataset, and the predictions that the model generated were validated with the help of the validation dataset. We conducted our experiment on the compiler Jupiter Notebook, which integrates with a number of packages and has an "Intel® CoreTM i7-10510U (1.8 GHz, up to 4.9 GHz, 8 MB cache, 4 cores)" powered by NVIDIA GeForce MX330 (2GB), 16GB DDR4 RAM, and a Windows 10 based 64 bit operating system. For training and testing data, we made use of the Google Colabratory platform. As a direct consequence of this, we carried out an analysis of the usefulness of our proposed model, which was constructed based on the forecasts that were produced by our trained model using the test dataset. To get a complete picture of the model's performance, we looked at its accuracy, precision, recall, and F1 score.

D. Real-Time User Interfcae:

The final step is to design a user interface (UI) that can recognize sign language and convert it to text. This was done using the Python OpenCV library. The palm motion image is gathered using the computer's webcam, then transformed to grayscale and scaled to 28x28 pixels using OpenCV tools. The preprocessed image was input to the pretrained model. When the model predicts a class for an image, the class is shown on the computer screen in the proper alphabet. The user or anyone else can see it. This phase takes the captured signed gesture as input and outputs the associated alphabet or image as text. Fig. 3 shows the application's user interface.

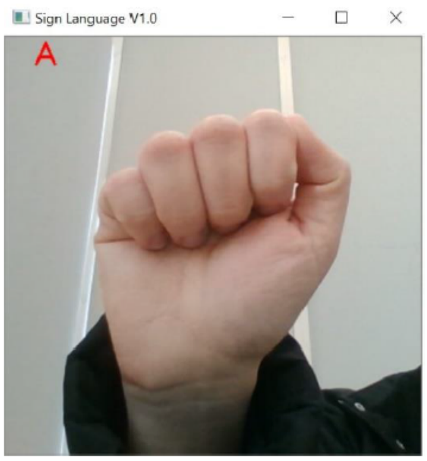

Fig. 3. UI of the Application

V. Evaluation Metrics

Recall, Precision, Accuracy, and F1 score are the metrics selected to monitoring the performance of our models. The mathematical expressions for these metrics are as follows:

$$Recall = \frac{TP}{TP + FN} \quad (5)$$

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

$$F1score = \frac{2 * Recall * Precision}{Recall + Precision} \tag{8}$$

VI. Results and Discussion

The following is the illustration of the performance attained after training and testing the models:

A. *Shallow Classifiers Performances:*

The accuracy obtained by the shallow classifiers is listed in Table I.

TABLE I
Shallow Classifiers Performances

Classifiers	Test Accuracy
Linear Regression	71.49%
Logistic Regression	90.16%
Random Forest	98.63%
Support Vector Machine (SVM)	97.92%

In this study, our necessity is for a probabilistic value for classification, but linear regression provides us with a continuous value, and the value is very much lower than the other classifiers. Using a random forest model, a response variable is calculated by generating a large number of decision trees and then running each model object through each tree to evaluate the results. This model, with the help of 100 decision trees, used averaging to improve the predictive accuracy and solve the problem of overfitting with an accuracy of 98.63%. For SVM, one-vs-one was chosen as the shape of the decision function due to the nature of the dataset being multiclass classification, and this classifier performed with 97.92% test accuracy.

B. *Performance of ANN and CNN:*

Following the training phase, the neural network was evaluated using the testing dataset, which consisted of 7,712 samples that the network had not previously encountered during the training phase. Performance evaluations, including accuracy, precision, recall, and F1 scores, were measured after they were run on the testing dataset, the results are summarized in Table II.

The accuracy and loss graphs for training and validation accuracy and loss are shown in Fig. 4 and Fig. 5, respectively. It was also possible to obtain a confusion matrix in addition to the performance metrics that were computed and given in Table II. After the model has been run, a confusion matrix is used to show the genuine positive and negative impacts that occurred. A comprehensive grasp of the flawed model, as well as the number of genuine negatives and false positives, is gained as a result of this procedure. The confusion matrix aids in the derivation of the conclusion that the majority of the predictions provided by the model are right. Figure 6 depicts a confusion matrix for the situation. However, due to the similarity of multiple letters in the ANN model, it is possible to detect substantial flaws in the model. Some inaccurate predictions are caused, for example, by the similarity between the locations of the fingers for A and S. Our suggested CNN model is very good at recognizing American Sign Language, so it gives very good results when it comes to recognizing this type of language, too.

TABLE II
ANN and CNN Model Classification Results

Model	Precision	Recall	F1 Score	Test Accuracy
ANN	0.82	0.83	0.81	82.96%
CNN	1.00	1.00	1.00	100%

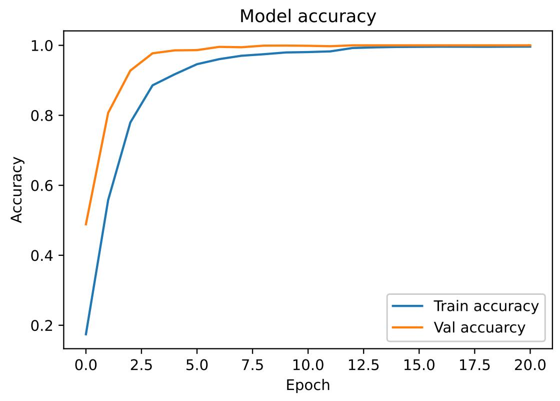

Fig. 4. Proposed Model Accuracy

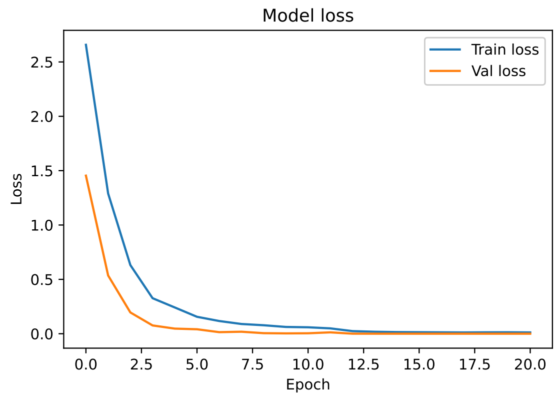

Fig. 5. Proposed Model Loss

Table III demonstrates that our suggested CNN model outperforms the other existing approaches in terms of accuracy and other metrics.

In comparison to the recent studies on sign language recognition, our research yielded a very high accuracy. The positive side of our research is flexibility. Each time you want to add

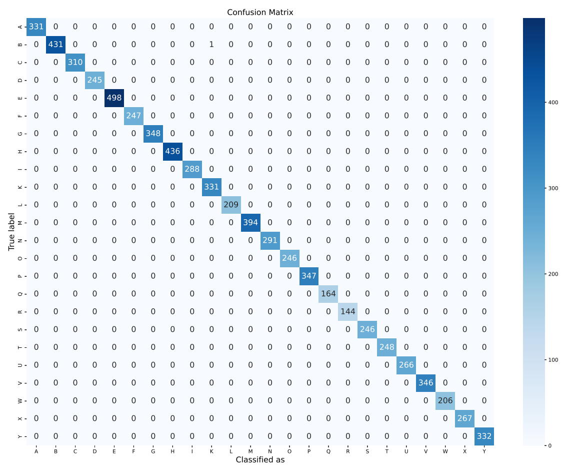

Fig. 6. Confusion Matrix

TABLE III
COMPARISON WITH PREVIOUS WORKS

Authors	Methods	Accuracy(%)
A. S. Konwar *et al.* [5]	HSV and Edge detection	65%
S. Upendran *et al.* [6]	PCA and KNN	77.29%
V. Bheda *et al.* [19]	DCNN	82.5%
Srinath S*et al.* [7]	Segmentation technique	86.67%
J. R. Pansare *et al.* [8]	Edge Orientation Histogram	88.26%
G. A. Rao *et al.* [10]	CNN	92.88%
Rathi *et al.* [11]	Transfer Learning	95.06%
Singha *et al.* [9]	Weighted Euclidean Distance	97%
Proposed	**CNN**	**100%**

new gestures, just add images of the gesture and train again; no need to use gloves with sensors, just use bare hands. But there are some things we can't do: we can't recognize dynamic gestures, we can't recognize gestures at night or in flare light, and the hand's oriented histogram still makes it hard to tell the difference between gestures that are almost the same.

VII. CONCLUSION

Many people use sign language, and computer vision could help them communicate more effectively. ASL is the most commonly used sign language worldwide. However, due to its complexity and wide range of classes, American Sign Language is difficult to understand using computer vision. This research presents a method for recognizing ASL alphabets using CNN. Our approach varies from others that rely heavily on transfer learning. Because of the similarities in hand movements, letters like "J" and "Z" were trimmed from the classification. The findings show that the proposed strategy achieved excellent classification performance. The proposed CNN model has a benchmark accuracy rate of 100% on our test dataset. Use the proposed CNN model with a hardware-based system that can read signs faster to improve performance.

In the future, this work can certainly be upgraded. Recognizing the alphabet is just the first step. By extending the field of view from the palm/hand to the posture of the body and head, the facial expression could be upgraded to words, which will make communication even easier.

ACKNOWLEDGMENT

The authors thankfully wish to acknowledge the authority for preparing the image database of the ASL alphabet and the Google Colab authority for providing the GPU services.

REFERENCES

[1] M. E. Al-Ahdal and M. T. Nooritawati, "Review in sign language recognition systems," in 2012 IEEE Symposium on Computers & Informatics (ISCI). IEEE, 2012, pp. 52–57.
[2] S. Y. Kim, H. G. Han, J. W. Kim, S. Lee, and T. W. Kim, "A hand gesture recognition sensor using reflected impulses," IEEE Sensors Journal, vol. 17, no. 10, pp. 2975–2976, 2017.
[3] H. Yi, S. Shiyu, D. Xiusheng, and C. Zhigang, "A study on deep neural networks framework," in 2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC). IEEE, 2016, pp. 1519–1522.
[4] M. A. Islam, "Reduced dataset neural network model for manuscript character recognition," 2020.
[5] A. S. Konwar, B. S. Borah, and C. Tuithung, "An american sign language detection system using hsv color model and edge detection," in 2014 International Conference on Communication and Signal Processing. IEEE, 2014, pp. 743–747.
[6] S. Upendran and A. Thamizharasi, "American sign language interpreter system for deaf and dumb individuals," in 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT). IEEE, 2014, pp. 1477–1481.
[7] S. Srinath and G. K. Sharma, "Classification approach for sign language recognition," in International Conference on Signal, Image Processing, Communication & Automation, 2017, pp. 141–148.
[8] J. R. Pansare and M. Ingle, "Vision-based approach for american sign language recognition using edge orientation histogram," in 2016 International Conference on Image, Vision and Computing (ICIVC). IEEE, 2016, pp. 86–90.
[9] J. Singha and K. Das, "Recognition of indian sign language in live video," arXiv preprint arXiv:1306.1301, 2013.
[10] G. A. Rao, K. Syamala, P. Kishore, and A. Sastry, "Deep convolutional neural networks for sign language recognition," in 2018 Conference on Signal Processing And Communication Engineering Systems (SPACES). IEEE, 2018, pp. 194–197.
[11] D. Rathi, "Optimization of transfer learning for sign language recognition targeting mobile platform," arXiv preprint arXiv:1805.06618, 2018.
[12] TECPERSON, "Sign language mnist," [Online]. Available: https://www.kaggle.com/datasets/datamunge/sign-language-mnist.
[13] H. Zhao, Z. Xu, G. Han, and D. Xie, "The fast image recognition based on grayscale features," in 2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE), vol. 3. IEEE, 2012, pp. 730–734.
[14] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," in 2018 international interdisciplinary PhD workshop (IIPhDW). IEEE, 2018, pp. 117–122.
[15] A. F. Agarap, "Deep learning using rectified linear units (relu)," arXiv preprint arXiv:1803.08375, 2018.
[16] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in 2017 international conference on engineering and technology (ICET). Ieee, 2017, pp. 1–6.
[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.
[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," The journal of machine learning research, vol. 15, no. 1, pp. 1929–1958, 2014.
[19] V. Bheda and D. Radpour, "Using deep convolutional networks for gesture recognition in american sign language," arXiv preprint arXiv:1710.06836, 2017.