Sign Language Recognition and Text to Speech Conversion

Md. Kausar Islam Bidhan
md.kausar.islam.bidhan@g.bracu.ac.bd

Sadia Yesmin
sadia.yesmin@g.bracu.ac.bd

Sifat E Jahan
sifat.jahan@g.bracu.ac.bd

Annajiat Alim Rasel
annajiat@gmail.com

***Abstract*—The vast majority do not understand sign language. It is challenging to find experienced interpreters. This work tries to develop a system-based real-time model on the Neural Networks for ASL finger writing. The hand is filtered first in this process to identify the category to which the hand gestures belong before being subjected to a classifier. This approach yields a 95.7% success rate for all 26 letters.**

***Index Terms*—components, configuring, style, styling, insert**

I. Introduction

American sign language is the most widely utilized of all the sign languages that are now in use. Sign language is the only form of communication available to those with deaf and mute (DM), who only have a communication impairment. The act of conveying information from one person to another through a variety of media, including words, gestures, body language, and pictures, is referred to as communication. People that are DM frequently utilize hand gestures to convey their messages. Gestures are used in nonverbal communication and can be visually interpreted. Sign language is used by the deaf and the mute to communicate non-verbally. The major goal of this work is to create a model that can identify hand motions based on finger spelling and combine them to create a whole word.

II. Motivation

Due to the differences between sign language and standard text, normal people and DM persons cannot communicate with one another. As a result, they are only able to communicate through their eyes. A study has been carried out on interface systems to validate DM persons' ability to communicate with one another without truly understanding one another languages. The goal is to develop appealing human-computer interfaces (HCI) that make it possible for computers to understand sign language. Worldwide, there are several sign languages in use, each with distinct syntax and features. French Sign Language (FSL), British Sign Language (BSL), Indian Sign Language (ISL), and Japanese Sign Language (JSL) are a few examples. In various corners of the world, there are also ongoing efforts to create new sign languages.

III. Methodology

The ultimate goal determines how the system functions. Every indication is made by hand, therefore communication doesn't involve the employment of any technical intermediaries.

A. The Generation of Data Sets

During the course of the project's development, it looked for pre-existing datasets but was unable to locate any raw image datasets that satisfied the needs of the task at hand. The datasets as RGB values are found. As a result, a random set of numbers is used here. The steps for gathering the data set are as follows. The dataset was produced with the aid of the OpenCV library. Each ASL symbol was first photographed approximately 800 times for training purposes and over 200 times for testing. Initially, record each and every image that appears on the webcam. In each of the images below, an ROI has been highlighted in blue to indicate the area of focus for this study. The illustration that follows demonstrates how the ROI is extracted from the RGB original before it is transformed into a grayscale image. The image is subsequently processed using the Gaussian Blur filter, which helps feature extraction. Here is what the image looks like after being blurred using the Gaussian method.

B. Classification of Gestures

- ***This is the strategy employed in this project:*** The method uses two different levels of algorithms for forecasting the user's concluding symbol.

- ***Algorithm Layer 1:***
  1) The image is processed by first adding a threshold and Gaussian blur filter after the features are separated in Open CV.
  2) A convolution neural network model is given the modified image for prediction.
- ***Algorithm Layer 2:***
  1) Many groups of signs are detected with consistent perception results.
  2) In order to categorize data amongst various subsets, the model uses classifiers created especially for those subsets.
- ***1st Layer***
  1) The first convolution layer must cope with the 128x128 input image. The first convolution layer uses the data with 32 filter-out parameters (3x3 pixels respectively).

2) ***1st Layer :*** The images are pooling down which means that the value in each 3x3 is retained. This results in a reduced resolution of 63x63 pixels for our image.
3) ***2nd Convolution Layer :*** The output of the first pooling layer, a 63-by-63 matrix, is used as input for the third layer, a convolution neural network.
4) ***Densely Connected Layer :*** In order to feed the first densely linked layer with its output from the second convolution layer, which initially receives the input images and converts them into a collection of 30x32x30 numbers. This layer receives a 28800-valued array in total. The output from this layer is delivered to the second layer which is closely coupled
5) ***Final layer :*** The final layer receives its input from the output of the second densely connected layer, and its number of neurons is equal to the classes that were being classified (including the alphabets and blank symbols).

- ***The Activation Function :***
  Here, in each layer, there implemented the ReLU. Each input pixel is given a max(x,0) calculation by the ReLu. This increases the formula's nonlinearity and helps students learn more complicated features. By requiring less computing time, it contributes to eradicating the issue of disappearing gradients and expedites the training process.
- ***The Pooling Layer:***
  On the image that was given to us, the Max-Pooling algorithm was used with the ReLU Activation Function and the pool size set at (3, 3). This causes the overall number of parameters to drop, which in turn causes the cost of computation as a whole to fall and overfitting to decrease.
- ***Dropout Layers:***
  An activation subset from the layer before is arbitrarily zeroed out in this layer. The network should classify or output a certain example even if some activation is removed[1].
- ***Dropout Layers:***
  The Adam optimizer changes the model based on the outcomes of the lower performance. Adam integrates the advantages of the adaptive gradient algorithm's (ADA GRAD) and root mean square propagation's (RMSProp) stochastic gradient descent method expansions.
- ***Layer 2***
  In order to as closely as possible identify the symbol that is being displayed, a two-layer strategy is utilised to forecast and verify symbols that are most similar to one another. Following symbols were displayed improperly and other symbols were displayed in their stead, according to testing:

  1) For R: U and D
  2) For U: D and R
  3) For S: U, T,K, and S
  4) For I: M and N

  As a result, we developed three distinct classifiers to handle the aforementioned scenarios:

  1) U, R,D
  2) T,K, U,S
  3) I, M,N
- ***Constructing a statement with only the fingertips:***
  1) **The Implementation:**
     Here, the letter that surpasses the predetermined threshold is demonstrated, and it is added to the existing string. In that case, the possibility of an inaccurate letter is removed from the present dictionary, and the number of current symbol detection. A blank is defined as an area with no pattern or texture. In any other scenario, it prints a space after the word it thinks will be the last one, and the current sentence is appended to the one that comes after it.

  2) **Auto correct Feature :**
     For each (incorrect) input word, a Python module called Hunspell proposes is used to recommend appropriate alternatives. The user is then presented with a list of alternatives that are compatible with the current word, from which he or she can select a word to add to the existing statement. If the user chooses a word, it will be added to the current sentence if it matches the existing term. Thus, fewer spelling mistakes are made, and it also makes it easier to anticipate challenging terms.

C. Training and Testing :

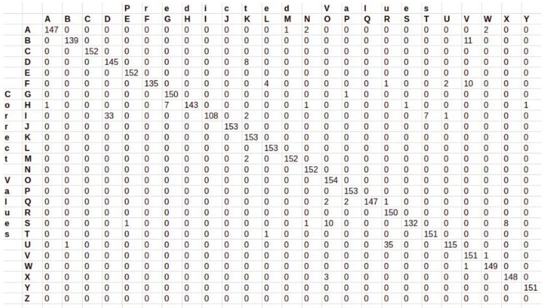
To filter out unwanted details, a Gaussian blur to the grayscale input images is applied after converting them from RGB. The images are resized to 128x128, and an adaptive threshold is used to separate the hand from the background. The input photos should then be subjected to all of the aforementioned actions before being fed into the model for training and testing. The prediction layer guesses correctly which category the image falls within. This means that the output is scaled. This was made possible in great part by the soft-max function. The prediction layer's output will initially vary significantly from the actual value. By putting networks through training on labeled data, the project made improvements. When classifying data, Cross-Entropy is used as a performance statistic. It is a continuous function that is exactly zero unless it reaches its labeled value, at which is non-negative. Therefore, by adjusting the weights of the neural networks within the network layer, the cross-entropy was as close to zero as possible. The cross-entropy can be easily computed with a built-in function in TensorFlow. After determining the cross entropy function, Gradient Descent is used to fine-tune it; in particular, Adam Optimizer is used here.

IV. Challenges Faced :

In this project, there are encountered a number of obstacles. The lack of a complete data set was the first challenge encountered. Since it was much more manageable to work with only square images than to deal with raw images as CNN does in Keras, As there are not any suitable data sets, choosing which filter to employ on the photos to extract useful characteristics and utilize the generated image as input for the CNN model presented the second challenge. There were a number of filters evaluated, such as binary threshold and canny edge detection, before choosing Gaussian blur.

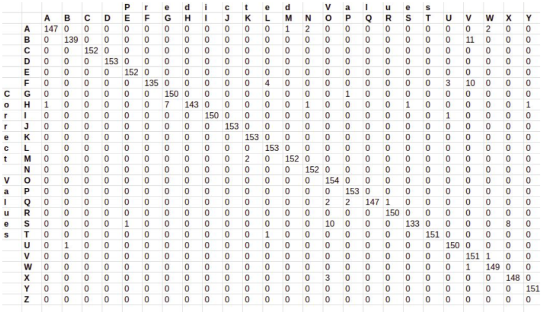
V. Results :

This model achieves an accuracy of 98.0% when layers 1 and 2 are combined, which is greater than the accuracy of the majority of the recent research articles on American Sign Language. It also achieves an accuracy of 95.8% with just layer 1 of the method. The vast majority of these studies examine the effectiveness of hand-detection hardware like the Kinect. [2] describes a system that uses Kinect and convolution neural networks to recognize Flemish sign language with a 2.5 percent mistake rate. [3] shows how to create a recognition model using a hidden Markov model classifier with a 30-word vocabulary and a 10.90



						P	r	e	d	i	c	t	e	d		V	a	l	u	e	s					
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
	A	147	0	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	2	0	0
	B	0	139	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0
	C	0	0	152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	D	0	0	0	145	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	E	0	0	0	0	152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	F	0	0	0	0	0	135	0	0	0	0	0	4	0	0	0	0	0	1	0	0	2	10	0	0	0
C	G	0	0	0	0	0	0	150	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
o	H	1	0	0	0	0	0	7	143	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1
r	I	0	0	0	33	0	0	0	0	108	0	2	0	0	0	0	0	0	0	0	7	1	0	0	0	0
r	J	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e	K	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c	L	0	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0
t	M	0	0	0	0	0	0	0	0	0	0	2	0	152	0	0	0	0	0	0	0	0	0	0	0	0
	N	0	0	0	0	0	0	0	0	0	0	0	0	0	152	0	0	0	0	0	0	0	0	0	0	0
V	O	0	0	0	0	0	0	0	0	0	0	0	0	0	0	154	0	0	0	0	0	0	0	0	0	0
a	P	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0
l	Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	147	1	0	0	0	0	0	0	0
u	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	150	0	0	0	0	0	0	0
e	S	0	0	0	0	1	0	0	0	0	0	0	0	0	1	10	0	0	0	132	0	0	0	0	8	0
s	T	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	151	0	0	0	0	0
	U	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	35	0	0	115	0	0	0	0
	V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	151	1	0	0
	W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	149	0	0
	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	148	0
	Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	151
	Z	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 1. **Algorithm 1**



						P	r	e	d	i	c	t	e	d		V	a	l	u	e	s					
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
	A	147	0	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	2	0	0
	B	0	139	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0
	C	0	0	152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	D	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	E	0	0	0	0	152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	F	0	0	0	0	0	135	0	0	0	0	0	4	0	0	0	0	0	0	0	0	3	10	0	0	0
C	G	0	0	0	0	0	0	150	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
o	H	1	0	0	0	0	0	7	143	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1
r	I	0	0	0	0	0	0	0	0	150	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
r	J	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e	K	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c	L	0	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0
t	M	0	0	0	0	0	0	0	0	0	0	2	0	152	0	0	0	0	0	0	0	0	0	0	0	0
	N	0	0	0	0	0	0	0	0	0	0	0	0	0	152	0	0	0	0	0	0	0	0	0	0	0
V	O	0	0	0	0	0	0	0	0	0	0	0	0	0	0	154	0	0	0	0	0	0	0	0	0	0
a	P	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0
l	Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	147	1	0	0	0	0	0	0	0
u	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	150	0	0	0	0	0	0	0
e	S	0	0	0	0	1	0	0	0	0	0	0	0	0	0	10	0	0	0	133	0	0	0	0	8	0
s	T	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	151	0	0	0	0	0
	U	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	150	0	0	0	0
	V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	151	1	0	0
	W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	149	0	0
	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	148	0
	Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	151
	Z	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 2. **Algorithm 1 + Algorithm 2**

VI. Future Scope :

By experimenting with a variety of algorithms for background subtraction, a level of accuracy can be achieved regardless of the complexity of the background. Additionally, there is scope for contemplating the possibility of enhancing the preprocessing in order to improve the efficiency with which gestures can be predicted in low-light situations.

VII. Conclusion :

This report's objective is to provide an overview of the creation of a useful ASL Sign Language DM detection system based on immediate time vision. It ultimately achieved a 98.0 percent accuracy rate for this dataset. By doing so, it is now possible to produce more precise predictions because of the use of two layers of algorithms that will verify and forecast symbols that are more similar to one another. It is possible to recognize all the symbols nearly by doing this and there is enough light.

References

[1] Aeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convol utional-Neural-Networks-Part-2/

[2] Pigou L., Dieleman S., Kindermans PJ., Schrauwen B. (2015) Sign Language Recognition Using Convolutional Neural Networks. In: Agapito L., Bronstein M., Rother C. (eds) Computer Vision - ECCV 2014 Workshops. ECCV 2014. Lecture Notes in Computer Science, vol 8925. Springer, Cham.

[3] Zaki, M.M., Shaheen, S.I.: Sign language recognition using a combination of new vision-based features. Pattern Recognition Letters 32(4), 572–577 (2011).

[4] N. Mukai, N. Harada, and Y. Chang: Japanese Fingerspelling Recognition Based on Classification Tree and Machine Learning. Kyoto, Japan, 2017, pp. 19-24. doi:10.1109/NICOInt.2017.9.

[5] Byeongkeun Kang , Subarna Tripathi, Truong Q. Nguyen " Real-time sign language fingerspelling recognition using convolutional neural networks from depth map" 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR).