SIGN LANGUAGE RECOGNITION USING TRANSFER LEARING

BY

S.M SHOHANUR HOSSAIN SOURAV
ID: 181-15-1965

SUMAIYA REZA
ID: 181-15-1752

This report is presented in partial compliance with the Qualifications Requirements for Computer Science and Engineering.

Supervised By

Md. Mahfujur Rahman
Senior Lecturer
Department of CSE
Daffodil International University

Co-Supervised By

Al Amin Biswas
Lecturer
Department of CSE
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

JANUARY 2022

APPROVAL

The project entitled **“Sign Language Recognition Using Transfer Learning”**, submitted by **S.M Shohanur Hossain Sourav, ID No: 181-15-1965 and Sumaiya Reza, ID No: 181-15-1752** at the Department of Computer Science and Engineering, Daffodil International University, has been well received. To meet part of the requirements for the B.Sc degree. in Computer Science and Engineering and approved its style and content. Presentation made 17th January.

BOARD OF EXAMINERS

[illegible]

Professor Dr. Touhid Bhuiyan **Chairman**
Professor & Head
Department of CSE
Faculty of Science & Information Technology
Daffodil International University

[illegible]

Dr. Md. Ismail Jabiullah
Professor **Internal Examiner**
Department of CSE
Faculty of Science & Information Technology
Daffodil International University

[illegible]

Narayan Ranjan Chakraborty
Assistant Professor **Internal Examiner**
Department of CSE
Faculty of Science & Information Technology
Daffodil International University

[illegible]

____

Dr. Mohammad Shorif Uddin **External Examiner**
Professor
Department of CSE
Jahangirnagar University

DECLARATION

We therefore make declaration that this work has been done by us under the watchful eye of **Md. Mahfujur Rahman,** a Senior Lecturer in the Department of CSE Daffodil International University. We also announce that neither this project nor any part of this project has been relocated to be awarded any degree or diploma.

Supervised by:

[illegible]
15.1.2022

Md. Mahfujur Rahman
Senior Lecturer
Department of CSE
Daffodil International University

Co-Supervised by:

[illegible]

Al Amin Biswas
Lecturer
Department of CSE
Daffodil International University

Submitted by:

S.M Shohanur Hossain Sourav

S.M Shohanur Hossain Sourav
ID: 181-15-1965
Department of CSE
Daffodil International University

Sumaiya Reza

Sumaiya Reza
ID: 181-15-1752
Department of CSE
Daffodil International University

ACKNOWLEDGEMENT

First of all we express our deepest gratitude and gratitude to Almighty Allah for His divine blessing enabling us to successfully complete our final year research based project.

We are very grateful and wish our deepest debt to **Md. Mahfujur Rahman, Senior Lecturer,** Department of CSE Daffodil International University, Dhaka. In-depth knowledge and in-depth interest of our manager in the field of “Machine Learning” to undertake this project. His unwavering patience, expert guidance, constant encouragement, unwavering supervision and enthusiasm, constructive criticism, valuable advice, a lot of low draft learning and correction at all stages made it possible to complete the project.

We would like to express our deepest gratitude to our Parents, our Family, and the Head of the CSE Department “**Professor Dr. Touhid Bhuiyan”**, for his kind assistance in completing our project and for the other members of the faculty and staff of the CSE department of Daffodil International University.

We would like to thank the entire study of our partner at Daffodil International University, who participated in this discussion while completing the course work.

Finally, we must respectfully acknowledge the support and support of our parents' patients.

ABSTRACT

Sign language (SL) is a visual language that people with speech and hearing disabilities use to communicate in their everyday conversations. It is entirely an optical communication language due to its native grammar. Sadly, learning and practicing sign language is not that common in our society; as a result, this research created a prototype for sign language recognition. Hand detection was used to create a system that will serve as a learning tool for sign language beginners. In this project work, we have created a improved Deep CNN model that can recognize which letter, word or digit of the American Sign Language (ASL) is being signed from an image of a signing hand [1]. We have extracted the features from the images by using Transfer Learning and build a model using Deep Convolutional Neural Network or Deep CNN. We used Tensorflow and Keras as a framework for our project. We have evaluated the proposed model with our custom dataset as well as with an existing dataset. Our improved Deep CNN model gives only an 4.95% error rate. In addition, we have compared the improved Deep CNN model with other traditional methods and here the improved Deep CNN model achieved an accuracy rate of 95 % and outperforms the other models.

TABLE OF CONTENTS

LIST OF FIGURES

LISTS OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Introduction

Communication is very important to the formation of a nation. Better communication leads to better understanding, which benefits everyone in the community, including the deaf and people with speech problem. People can communicate with one another using sign language. The majority of hearing people, on the other hand, have no understanding of sign language, and learning it is difficult. There is not enough study done yet related to sign language. As a consequence, there is yet a significant gap between the hearing impaired and the hearing majority.

There are different sign languages used by speech and hearing-impaired people from everywhere in the globe, including American Sign Language, Indian Sign Language, Australian Sign Language, Italian Sign Language etc. [2]. ASL is the most widely used sign language in the world. ASL is used not only in the United States, but also in Canada, Mexico, West Africa, and Asia. More than 20 other countries, including Jamaica, Panama, Thailand, and Malaysia, use ASL to communicate with their hearing-impaired communities [3].

The major object of our Sign Language Recognition System is to provide a fast and precise way for the hearing-impaired community to recognize the sign language more accurately. To get the desired result we built an improved Deep CNN model. To obtain data from the signer, we used Google drive. Our datasets were uploaded there as zip files and we used unzip command in Google Colab to extract them. The system's goal is to serve as a learning tool for those who is interested in learning additional information about the fundamentals of sign language, such as alphabets, words, and digits. We have used Deep CNN model and VGG16 pre-trained model at first. We then combined the idea of the both model and used transfer learning. Our improved Deep CNN model gives a better accuracy than the other models.

1.2 Motivation

Deaf and hard of hearing people can communicate with the rest of society with the help of a good Sign Language Recognition system. The objective of Sign Language Recognition (SLR) is to create methods and methodologies for correctly recognizing a succession of gestures and understanding their meaning. Sign Language Recognition is a tenacious and motivating task due to numerous constraints and factors. Sign Language Recognition is a notable task because of its impact on the society, as speech and hearing-impaired peoples face a significant communication gap with the speaking community. We wanted to help them to overcome with this communication gap and as result we have come up with this research project. There are numerous opportunities to conduct research on recognizing American Sign Language in order to improve communication between the mute and speaking communities. Hand gestures in Sign Language are used in text or speech translation systems or the alternatives for the speech and hearing-impaired people that are used in public places like airports, post offices, or hospitals.

1.3 Rationale of the Study

This study was conducted to identify sign language using images from a dataset and learn how to classify any hand gesture is signing. We have trained our model to identify alphabets, words and digits. We have worked on this research to help the deaf/mute people as well as the hearing society. Researcher will get help from our study that which methodology gives faster output and better accuracy.

1.4 Research Questions

- How can we identify the alphabets?
- How can we identify the words?
- How can we identify the digits?
- What dataset do we use?
- What methodology do we use?
- What framework will we use?

1.5 Main Objectives

As our research goal is to identify sign language, we have used Deep CNN, transfer learning and transfer learning based improved Deep CNN model to recognize the sign language.

- To propose an improved Deep CNN that can easily identify a sign language.
- Identify the alphabets or words a person is signing.
- Identify the digits a person is signing.
- Using this system to reduce the workload and time needed to identify a sign language that can be used in other tasks instead.

1.6 Report Layout

The following is how we organized our project report:

- Chapter One contains the introduction of our research, motivation for this study, objectives, and expected outcome of this study.
- Chapter Two includes the background of the research, related works, research summary and challenges faced during this study.
- Chapter Three includes research methodology, the proposed systems, datasets, the implementation procedure, data preprocessing and the improved model.
- Chapter Four includes Experimental Results and Discussion including experimental setup, confusion matrix, performance and comparative analysis.
- Chapter five includes the Conclusion and Future Scope of this project.
- Lastly the References section contains the resources used during the research.

CHAPTER 2
BACKGROUND

2.1 Introduction

The age of information is the current phase of our civilization, and the computer and the internet are the main technological drivers behind it. We have used Deep CNN and VGG16 architecture and Transfer Learning in our research program. Deep CNN [14] is a type of specialized neural network that is used to process images. Deep CNN is usually used for image detection and classification. VGG is a model for recognizing objects that can support up to nineteen layers. VGG was created with the intention of being a deep CNN, outperforms baselines on a wide range of tasks and datasets outside of ImageNet. The practice of taking the knowledge or features of one problem and applying them to another problem is known as transfer learning [15]. It's like using software to detect cancerous tumor shapes instead of irregular bread shapes.

2.2 Related Works

We have already seen a lot of research on Deep Convolutional Neural Networks and Sign Language Recognition. We've looked over a few of these research papers and gotten some ideas from them.

Shivashankara S, Srinath S, et al. [1] proposed an ideal strategy with the main goal of transliterating 24 static American Sign Language alphabets and integers into humanoid or appliance comprehensible English composition. They found out majority voting was the most accurate out of a group consisting of GoogleNet, AlexNet, VGG-16, VGG-19, and ResNet-50. R. Elakkiya, et al. [2] proposed the impact of machine learning in the most up-to-date literature on sign language recognition and classification. This essay primarily focuses on resolving three key SLR concerns. That is to say elimination and sorting of strong subunit characteristics, stirring epenthesis activity and fulfilling the appearance for subunit sign modelling.

D. Forsyth, A. Farhadi and R. White et al. [3] proposed a technique to create word models for American Sign Language (ASL) that can be transferred between signers and aspects.

They use transfer learning for build this model. This method is demonstrated in two scenarios, from an avatar to a frontally seen human signer and from an avatar to a 3/4 view human signer. The primary objective of this study is showing the benefits of transfer learning using comparative characteristics. Lean Karlo S. Tolentino, Ronnie O. Serfa Juan et al. [4] proposed a system that convert static sign language into its verbal counterpart, which includes letters, digits, and fundamental phrases static indicators helping people understand the principles of Sign language is a kind of indication that is used to communicate. With regards to testing precision of 90.04 percent in letter acceptance, 93.44 percent in digit identification, and 97.52 percent in static word recognition, this model was capable of to reach 99 percent training accuracy, with mean of 93.667 percent based on posture recognition having a limited amount of time.

A.L.C. Barczak, N.H. Reyes et al. [5] narrated a fresh picture dataset is being created that can be utilized by additional researchers in computer vision. The first report of the training dataset includes 2425 pictures of 5 several people for each of the 36 ASL motions. Future report will have 18000 hands of 20 several people manifested in 5 several ways, with 5 repeats for every one of the 36 ASL motions. The dataset will be continually bring up to date with fresh pictures until it is complete. Md. Shahinur Alam, Mahib Tanvir, Dip Kumar Saha, Sajal K. Das et al. [6] proposed a model which perceive all the 36 letters and 10 digits of the BdSL with consequential accuracy. A sophisticated outcome with a validation accuracy of 99.57% and a validation loss of 0.56% has been execute in the recognition and interpretation of the BdSL symbol.

Teak-Wei Chong , Boon-Giin Lee et al. [7] This article describes an American Sign Language recognition system that employs the Controller for Leap allusion with 26 letters and 10 numbers. The study included a total of 23 characteristics, which were then split into six distinct groups of combinations. According to the findings, the space between two fingertips and the neighboring fingers is an important characteristic for sign language understanding. Anna Deza, Danial Hasan et al. [8] has proposed a neural network model which can determine the given picture of a signing hand of the American Sign Language (ASL) alphabet is being signed. This research is a preliminary step in developing a sign language translator that can take sign language messages and convert them into written and spoken language. This translator would substantially reduce the barrier for many mute people and who is unable to hear to converse with others in everyday situations.

J. Pansare and M. Ingle et al. [9] In the field of hand gesture analysis, they proposed two primary approaches, vision-based and device-based. They use Fourier Transformation to build this project. Using the significant of the Binary Linked Object (BLOB) technique, real-time HGRS has been created for the recognition of 26 static hand motions linked to the A-Z alphabets. This model probably recognizes single-hand movements in real-time and obtains a detection rate of 90.19%. Cheok, M.J., Omar, Z. & Jaward et al. [10] in this research paper they seeks to core on a survey of advanced methods. Although expression on the face is utilized in sign language, it is not covered in this work. This methods used for pre-processing, analysis, feature extraction, and distribution.

S. Upendran and A. Thamizharasi et al. [11] have proposed Deep Learning Computer Vision may be used to detect hand gestures by constructing Deep Neural Network designs (Convolution Neural Network Architectures) in which the model learns to recognize hand gesture photos throughout an epoch. When the model correctly detects the gesture, the matching English text is created, and the text may subsequently be translated to voice. The PCA characteristics in the ASL hand posture are extracted by this method. The collected PCA features may be utilized to categories the ASL alphabets with the k-NN classifier. Krizhevsky, A., Sutskever, I., Hinton, G.E et al. [12] have proposed Machine learning algorithms are used broadly in ongoing approaches to object recognition. To categories the 1.2 million high-resolution pictures in the ImageNet LSVRC-2010 challenge into the 1000 distinct frame, they trained a massive, deep convolutional neural network. They achieved top 1 and top 5 error rates of 37.5% and 17.0% commonly, on the test data, which is approximately is the superior way than the preceding advance.

Ivan Gruber, Dmitry Ryumin et al. [13] have proposed a convolutional neural network may be used to labeling numerical movements in sign language. They developed a fresh dataset of these motions for the sake of this study. The traditional VGG16 model was used for a distribution job, and the outcome were in comparison to the selected baseline technique and other examined architectures. They achieved a recognition accuracy of 86.45%, which is more than 34% better than the specified baseline approach.

2.3 Limitation of Existing Work

After studying the existing project works and papers we discovered that several alternative machine learning algorithms can be used to perform the sign language detection. These existing works has several limitations and we can fix that by our project. The most viable option for us is the open-source ImageNet like VGG16, Transfer learning and Deep Convolutional Neural Network that have publicly available documentation and codes. For this work, we also required to develop our own unique dataset. In that case our improved Deep CNN model will be very helpful to overcome the limitations.

2.4 Scope of The Problem

This project's purpose is to build a system for extracting features from images to identify alphabets, digits and words. The system also needs to extract some characteristics to identify if there is nothing in the picture. Our project was also labeled all the images.

2.5 Challenges

Data collection: We have faced so much trouble to customize our data. We had to take all the images for different hand gestures individually. We haven’t any idea of sign language hand gestures, how to express the signs. So, we have watched some tutorials and get helps from google how to express the sign languages. This took a big amount of time from us.

Data transfer: There are limitations to how much data can be stored from a mobile device as we have taken the images from our mobile. We also need to transfer the images from mobile to our computer via the USB cable. It took a lengthy moment to transfer all the images because we had a big number of images in our mobile and the cable was a mess.

Calculation limitations: There are limitations to how fast data can be processed and hand gestures detected. This can then cause a wrong sign language to be detected. We can rectify this by using cloud computing for our system.

Hardware limitations: There can be bottlenecks in our processing hardware where they will lag behind the captured images. This can easily be rectified by adding more robust hardware like CPU, HDD or GPU to our architecture. In fact, we have faced this problem in our training of the model where we had to leave our computers on for a few days.

Model limitations: From our research, we have seen that some models will underperform in either hand gesture recognition or sign language detection. We have tried to rectify this by using ImageNet and VGG16 models.

Weather and Light: The performance of the system can be severely impacted by adverse weather conditions. There may be limitations on identifying hand gestures in a low light environment like during the evening or night. That's why we had to take the pictures in a bright condition with a clear background.

CHAPTER 3
RESEARCH METHODOLOGY

3.1 Introduction

As our research goal is to detect hand gestures and identify sign language, we made use of Deep Convolutional Neural Network and Transfer Learning in order to solve that problem. We made use of DCNN or Deep Convolutional Neural Network algorithms which we have implemented using the Tensorflow framework. For training our dataset we have used a custom and existing dataset of images.

3.2 Proposed System

As our research goal is to identify sign language, we have used Deep CNN, transfer learning and transfer learning based improved Deep CNN model to recognize the sign language. Finally, we compared these three models and improved Deep CNN outperforms of the other models. At first, we have used a Deep CNN model with four different filters and used ReLU and Softmax activation layer. Next, we have used a pretrained model as VGG-16 to extract the features from images, train this pretrained model and get an accuracy. Finally, we build our transfer learning based improved Deep CNN model where pretrained model VGG-16 worked as a feature extractor and Deep CNN worked as a trainable model.

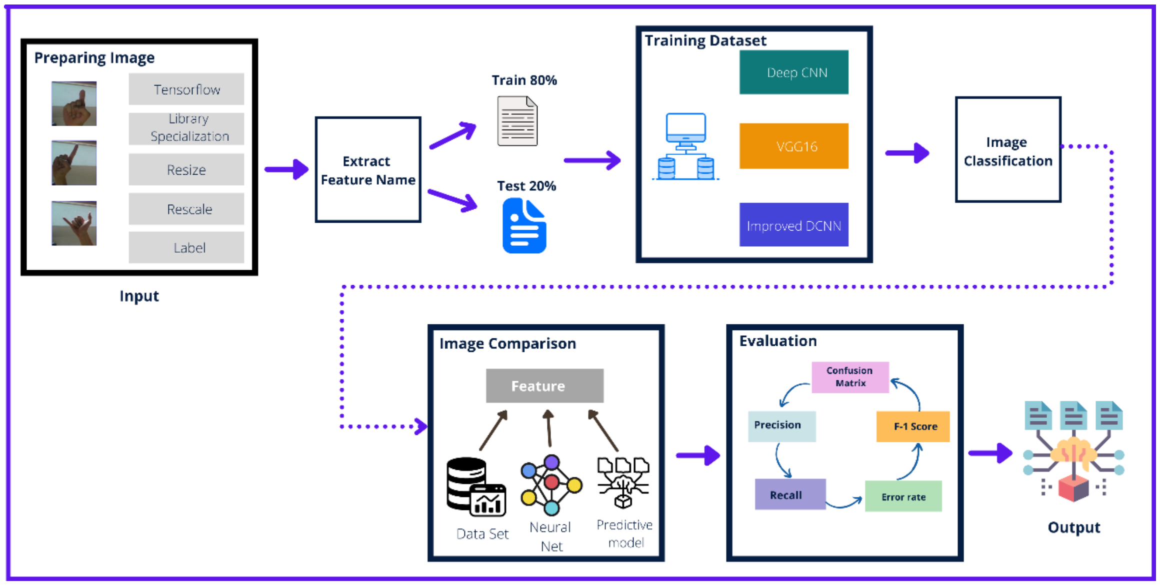

Figure 1: Working prototype of proposed methodology.

3.3 Dataset

The first step of implementing our model is collecting data. As our model requires an image dataset, we have taken images from google. The major source of data for this research was the ASL Alphabet, a collected dataset of American Sign Language (ASL) by Kaggle user Akash [19]. The collection is made up of 87,000 200x200 pixel pictures. There are 45 classes in all, 26 for the letters A-Z, 10 for the digits 0-9 and 8 for space, delete, nothing, dislike, ok, good job, home, please and others. We have also used our own custom dataset consist of 1000 pictures which were taken by our own mobile devices. And later on, we have also customized the training dataset with the downloaded dataset. We added 0-9 digits and some words in the training and testing datasets. Our raw images were high quality images and for that reason it took a bit long time for loading data.

3.4 Implementation Procedure

3.4.1 Data Preprocessing

Our dataset contains images of different sizes. Which will cause a problem while training it. So, in order to avoid that we have transformed and resized the images so that all the images are of the same size where we used image size of 50. Also, as we used Deep CNN model for training our dataset, it requires a lot of images. We had enough data to train our dataset. In the data pre processing part we have also labeled our images and used 3 channel color images (RGB). This helped us a lot to get better performance and quality images to train the dataset.

3.4.2 Deep Convolutional Neural Networks (DCNNs)

DCNNs, also known as ConvNets, are multi-layer neural networks that are primarily used for image processing and object detection. Multiple layers analyze and extract features from data in DCNNs.

Convolution Layer: DCNN includes a convolution layer with multiple filters to perform the convolution process.

Rectified Linear Unit (ReLU): A ReLU layer is used by DCNNs to execute operations on items. A corrected feature map is the result. When the flattened matrix of the pooling layer is supplied as an input, a fully connected layer appears, classifying and labelling the pictures.

Pooling Layer: After that, the feature map is corrected and sent to a pooling layer. Pooling is a technique for reducing the size of a feature map by down sampling it. The pooling layer converts the pooled feature map's two-dimensional arrays into a single linear vector that is long and continuous.

Fully Connected Layer: When the pooling layer's flattened matrix is used as an input, a fully connected layer come up that classifies and labels the images.

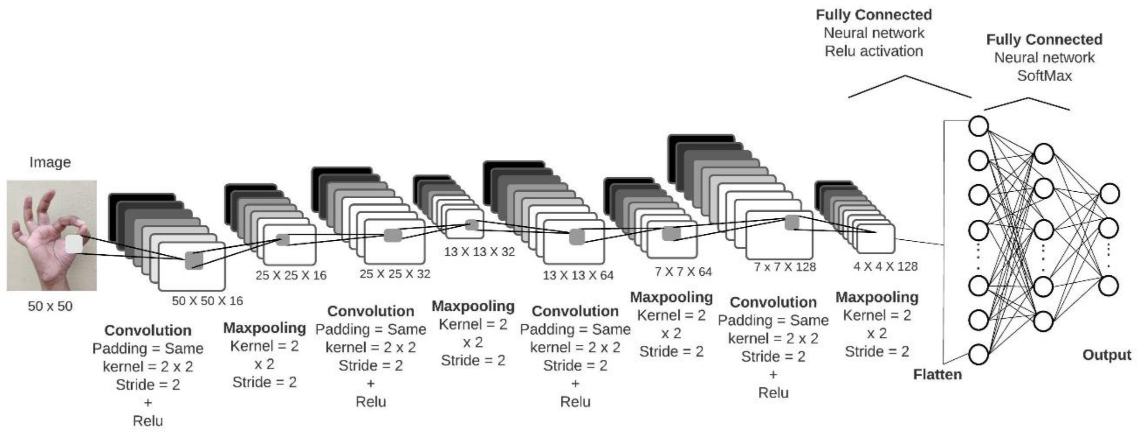

Figure 2: Working method of Deep CNN

3.4.2.1 Model Training

To develop our Deep CNN model, we have used four Convolution layer and two flatten layer. We used 16, 32, 64,128 filters in the four Convolution layer respectively. In this four layers we used kernel size of 2 and the activation function we used here is “Relu”. In the last dense layer we used “Softmax” classification algorithm. Here, we trained for 15 times with a batch size of 64. In this model we had 45 classes. On the validation set, the model is assessed at each epoch, and the model with the highest acceptance accuracy is saved to repository for future appraisal and use. Once the training is complete, the training and acceptance error and loss, as well as a plot of error and loss overtraining, are saved.
For improving our model, we have increased the batch size to 128. In the case of our dataset, we didn’t change any confusing classes. After reducing the epoch to 10, we get some great improvements in our model.

3.4.2.2 Model Evaluation

In contemplation of assess our model's performance, we made to use for confusion matrices (CM). From CM, we get four basic qualitative model quality indicators, namely, True positive (TP) which is genuinely positive, False positive (FP) which is not genuinely positive, True negative (TN) which is negative in the true sense and False negative which is negative that is not true (FN). Using the mentioned indicators, we have calculated the values of accuracy, f1-score (micro), precision (micro) of proposed model. Accuracy for a multi-class confusion matrix is the average percentage of valid forecasts. Moreover, Accuracy is a multi-class performance metric to evaluate the model. After comparing accuracy with the VGG16 model and Deep CNN model, our proposed transfer learning-based improved Deep CNN model outperformed the modified Deep CNN model and the VGG16 model.

TP (True Positive) = Correctly predicted classes.

FP (False Positive) = Anticipating negative classes in the act of positives.

TN (True Negative) = Predicting negative classes in the true sense.

FN (False Negative) = Forecasting negative classes that is not true.

Here, accuracy = (total TP + total TN) / (total TP + total TN +total FP + total FN)

F1-Score is the harmonic mean of precision and recall. For multiclass confusion matrices, we use micro f1-score.

Here, f1-score = (2*total TP / (2*total TP + total FP + total FN))

Precision means the portion of positive fragment which were correctly predicted in the act of positive. Here, we have used micro precision.

Here, precision = total TP / (total TP + total FN)

Error rate indicates the fraction of incorrect predictions.

Here, error rate = 1- accuracy.

We have created a table which named table 1 and we can observe the performance from this table on dataset which is trained by Deep CNN. Here are ten epoch shown and have five columns are Accuracy, Error rate, train loss, valid loss, and time. From this table we get the highest accuracy is 0.9445 and lowest accuracy is 0.5219.

In Table 1, though the accuracy was not bad, the model had huge validation loss. So, our model has rooms for improvement.

Table 1: Performance after training with Deep CNN

EPOCH	TRAIN LOSS	VALID LOSS	ACCURACY	ERROR RATE	TIME
0	1.5251	14.7713	0.5219	0.4781	01:08
1	0.4843	21.1318	0.8332	0.1668	01:07
2	0.2839	25.8563	0.9011	0.0989	01:07
3	0.1973	27.5158	0.9320	0.068	01:07
4	0.1539	33.3173	0.9376	0.0624	01:07
5	0.1294	33.6078	0.9400	0.0600	01:06
6	0.1051	36.5312	0.9412	0.0588	01:06
7	0.0913	35.4029	0.9423	0.0577	01:06
8	0.0878	35.6365	0.9438	0.0562	01:06
9	0.0775	40.9340	0.9445	0.0555	01:07

3.4.3 VGG16

VGG16 is a deep convolutional neural network which has 16 layers that's why it is called VGG16. It is possible to load a pre-trained variant of the network that has been trained on over a million photographs from the ImageNet database. The pre-trained network can categories photos into 1000 different Object classifications, including computers, hard disks, pens, and a wide range of creatures. The most unique feature of VGG16 is that, rather of a enormous number of hyper parameters. It has limited the number of filter in convolution layers to 3x3 with a stride of 1 and used the same padding every time. It also use maxpool layer of 2x2 filter and stride 2. We made to use for a pre-trained VGG16 model in our project to behold the performance of our data sets.

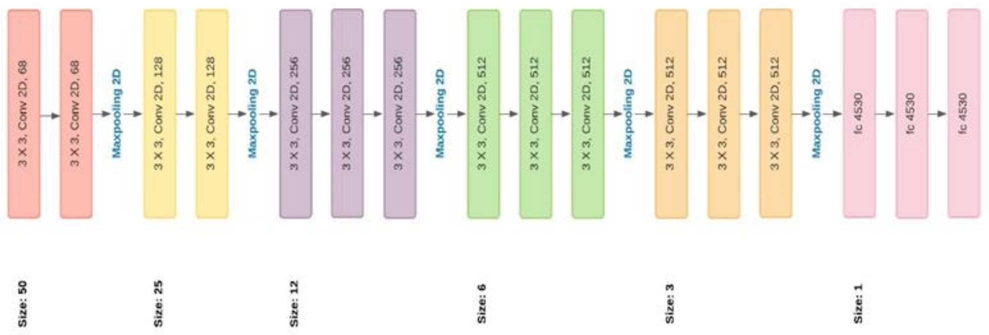

Figure 3: VGG16 model's architecture.

3.4.3.1 Model Evaluation

To appraise the performance of our model, we have used confusion matrices (CM). From CM, we get four basic qualitative model quality indicators, namely, True positive (TP) which is genuinely positive, False positive (FP) which is not genuinely positive, True negative (TN) which is negative in the true sense and False negative which is negative that is not true (FN). Using the mentioned indicators, we have calculated the values of accuracy, f1-score (micro), precision (micro) of proposed model. Accuracy for a multi-class confusion matrix is the average number of correct predictions. Moreover, Accuracy is a multi-class performance metric to evaluate the model. After comparing accuracy with the VGG16 model and Deep CNN model, our proposed transfer learning-based improved Deep CNN model outperformed the modified Deep CNN model and the VGG16 model.

TP (True Positive) = Correctly predicted classes.

FP (False Positive) = Anticipating negative classes in the act of positives.

TN (True Negative) = Predicting negative classes in the true sense.

FN (False Negative) = Forecasting negative classes that is not true.

Here, accuracy = (total TP + total TN) / (total TP + total TN +total FP + total FN)

F1-Score is the harmonic mean of precision and recall. For multiclass confusion matrices, we use micro f1-score.

Here, f1-score = (2*total TP / (2*total TP + total FP + total FN))

Precision means the portion of positive fragment which were correctly predicted in the act of positive. Here, we have used micro precision.

Here, precision = total TP / (total TP + total FN)

Error rate indicates the fraction of incorrect predictions.

Here, error rate = 1- accuracy.

We have created a table which named table 2 and from this table we can see the performance on dataset which is trained by VGG16. Here are ten epoch shown and have five columns are train loss, valid loss, Accuracy, Error rate and time. From this table we get the highest accuracy is 0.9113 and lowest accuracy is 0.5165 in epoch 0 and epoch 9. When our data set is trained with VGG16 a model of transfer learning, there have some loss value we got. The highest train loss happened in epoch 0 which is 1.5251 and the lowest train loss happened in epoch 9 which is 0.0775. A loss function is a metric that measures how well your forecasting model predicts the forecasting outcome (or value). The learning complication is recast as a development problem, with a loss function created and the approach was tweaked to reduce the loss function's size. Error, on the other hand, represents how well your network performs on a certain training, testing, validation set. A low mistake rate is desirable, whereas a large error rate is unquestionably undesirable. A loss function, of which there are numerous, is used to determine the error.

In Table 2, our training loss as well as our accuracy improved with other performance factors.

Table 2: Performance after training with VGG16

EPOCH	TRAIN LOSS	VALID LOSS	ACCURACY	ERROR RATE	TIME
0	2.2070	4.3844	0.5165	0.4835	18:13
1	1.3427	5.2367	0.7341	0.2659	18:10
2	1.0315	5.9215	0.7959	0.2041	18:11
3	0.8539	6.5137	0.8284	0.1716	18:08
4	0.7345	7.0395	0.8512	0.1488	18:08
5	0.6469	7.5203	0.8712	0.1288	17:53
6	0.5798	7.9606	0.8840	0.1160	17:54
7	0.5260	8.3708	0.8955	0.1045	17:53
8	0.4811	8.7582	0.9066	0.0934	17:56
9	0.4448	9.1174	0.9113	0.0887	17:53

For further improvement, we have optimized the parameters and changed the pretrained model to VGG16 (Without fine tuning). Even after reducing the epochs, we still get better results than Table 2.

3.4.4 Improved Deep CNN Model

Transfer learning is one kind of process of implementing a preceded learned model to a recent model. It is currently quite suitable in deep learning because it can train deep neural networks with relatively minimal data. Transfer learning occurs when a machine learning model that has already been trained is utilized to address a new but relevant complication. For instance, you could utilize the model's expertise to recognize additional things such as sunglasses if you trained a simple classifier to predict whether a shot contains a backpack. By transfer learning, we hope to implement what we have accomplished in one activity to improve abstraction in another. The substance learned by a network at "task A" are conveyed to a new "task B." We trained this transfer learning with our dataset. It's produced good accuracy. After that, for our proposed system we have used transfer learning to get the appearance from it. We have taken the output from the pre-trained model and added it with our Deep CNN model. It gave a good performance overall as the pre-trained VGG16 model's features transfers into the Deep CNN model. To create the transfer learning network we didn't train the layers of VGG-16 but extracted the features from it and we also add 3 custom layers with it. Here we have used the relu and softmax activation function with a dropout function of 0.2, 0.3.

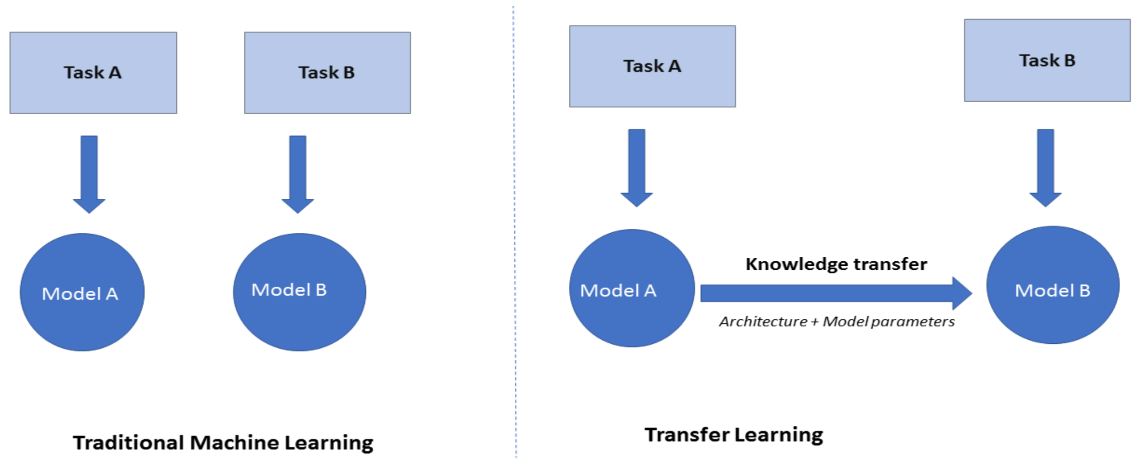

Figure 4: Working method of Transfer Learning.

For further improvement, we have optimized the parameters and changed the pre-trained model to VGG16 (Without fine tuning). Even after reducing the epochs, we still get better results than Table 2.

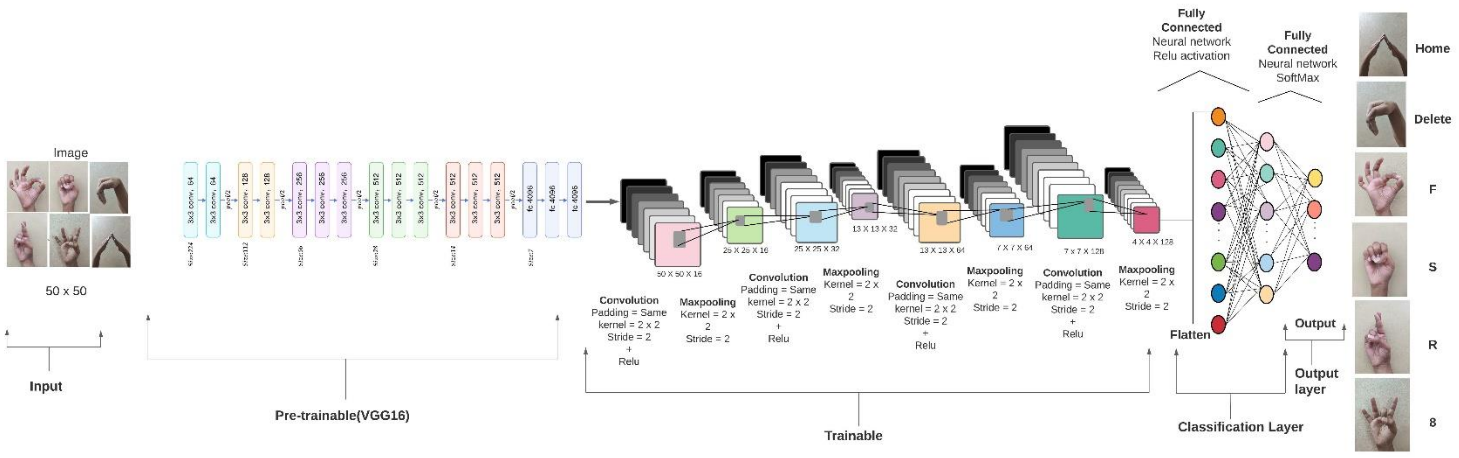
Image
50 x 50
Input
Pre-trainable(VGG16)
Convolution Padding = Same kernel = 2 x 2 Stride = 2 + Relu
Maxpooling Kernel = 2 x 2 Stride = 2
Convolution Padding = Same kernel = 2 x 2 Stride = 2 + Relu
Maxpooling Kernel = 2 x 2 Stride = 2
Convolution Padding = Same kernel = 2 x 2 Stride = 2 + Relu
Maxpooling Kernel = 2 x 2 Stride = 2
Convolution Padding = Same kernel = 2 x 2 Stride = 2 + Relu
Maxpooling Kernel = 2 x 2 Stride = 2
50 X 50 X 16
25 X 25 X 16
25 X 25 X 32
13 X 13 X 32
13 X 13 X 64
7 X 7 X 64
7 x 7 X 128
4 X 4 X 128
Trainable
Fully Connected Neural network Relu activation
Fully Connected Neural network SoftMax
Flatten
Output
Output layer
Classification Layer
Home
Delete
F
S
R
8

Figure 5: Work flow of Improved Deep CNN model.

As VGG16 requires much time for training, so it will require more GPU and ram. So, we might run out of memory. Currently almost every NVidia GPUs supports tensor cores which can speed up neural network training by 2x-3x. They also require less GPU memory.

3.4.4.1 Model Training

To train our proposed model, we have used a transfer learning methodology with some additional layers to train our improved Deep CNN model. We used the VGG-16 model to extract the features and added our Deep CNN model with it to get better output results. Here, we have used a batch size of 10 and trained for 15 epochs. Here we had 45 classes. The model is evaluated at every time on the acceptance set, and the model with the highest acceptance accuracy is saved to repository for future appraisal and use. Once the training is complete, the training and validation error and loss, as well as a plot of error and loss overtraining, are saved.

For improving our model, we have increased the batch size to 32. In the case of our dataset, we didn't change any confusing classes. After reducing the epoch to 10, we get some great improvements in our model.

3.4.4.2 Model Evaluation

To appraise the performance of our model, we have used confusion matrices (CM). From CM, we get four basic qualitative model quality indicators, namely, True positive (TP) which is genuinely positive, False positive (FP) which is not genuinely positive, True negative (TN) which is negative in the true sense and False negative which is negative that is not true (FN). Using the mentioned indicators, we have calculated the values of accuracy, f1-score (micro), precision (micro) of proposed model. Accuracy for a multi-class confusion matrix is the average number of correct predictions. Moreover, Accuracy is a multi-class performance metric to evaluate the model. After comparing accuracy with the VGG16 model and Deep CNN model, our proposed transfer learning-based improved Deep CNN model outperformed the modified Deep CNN model and the VGG16 model.

TP (True Positive) = Correctly predicted classes.

FP (False Positive) = Anticipating negative classes in the act of positives.

TN (True Negative) = Predicting negative classes in the true sense.

FN (False Negative) = Forecasting negative classes that is not true.

Here, accuracy = (total TP + total TN) / (total TP + total TN +total FP + total FN)

F1-Score is the harmonic mean of precision and recall. For multiclass confusion matrices, we use micro f1-score.

Here, f1-score = (2*total TP / (2*total TP + total FP + total FN))

Precision means the portion of positive fragment which were correctly predicted in the act of positive. Here, we have used micro precision.

Here, precision = total TP / (total TP + total FN)

Error rate indicates the fraction of incorrect predictions.

Here, error rate = 1- accuracy.

We produced a table called Table 3 and from it we can examine the performance on a dataset trained using Improved Deep CNN. There are 10 epochs displayed here, with five columns containing train loss, valid loss, accuracy, error rate, and time. From this table we get the highest accuracy is 0.95 and lowest accuracy is 0.87 in epoch 0 and epoch 9. When our data set is trained with Improved Deep CNN, there have some loss value we got. The highest train loss happened in epoch 0 which is 0.3687 and the lowest train loss happened in epoch 9 which is 0.0912. A loss function is a metric that measures how well your prediction model predicts the predicted outcome or value. The learning issue is transformed into an optimization problem, a loss function is defined, and the method is optimized to minimize the loss function. Error, on the other hand, represents how well your network performs on a certain training, testing, validation set. A low mistake rate is desirable, whereas a large error rate is unquestionably undesirable. A loss function, of which there are numerous, is used to determine the error.

Table 3: Performance of Improved Deep CNN

Epoch	Train loss	Valid loss	Accuracy	Error rate	time
0	0.3687	8.2469	0.8773	0.1227	18:13
1	0.2944	11.4947	0.9027	0.0973	18:10
2	0.2473	14.3312	0.9172	0.0828	18:11
3	0.2186	16.3780	0.9254	0.0746	18:08
4	0.1940	17.7813	0.9361	0.0639	18:08
5	0.1695	29.9138	0.9445	0.0555	17:53
6	0.1580	20.6153	0.9480	0.0520	17:54
7	0.1391	22.2125	0.9494	0.0506	17:53
8	0.1318	23.1417	0.9500	0.0500	17:56
9	0.0912	24.9417	0.9505	0.0495	17:53

In Table 3, our training loss as well as our accuracy improved with other performance factors.

CHAPTER 4
RESULTS AND DISCUSSIONS

4.1 Experimental Environment

The experiment is conducted on a personal computer where the microprocessor is 3.00 GHz Intel(R) Core(TM) i5-7400 CPU with NVidia GT-1030, 8 GB ram and Windows 10 operating system. The model is executed on an open-source platform Google Colab Notebook. The model is implemented on the Tensorflow and Keras framework.

4.2 Parameter Optimization

During the simulation of our proposed model, we concentrated on a few parameters in order to improve its performance. Here, we have increased the batch size to 64. It increased our model performance. Though we used a learning rate between 1e-5 and 1e-4, it didn’t improve our model performance. Even after trying up to 20 epochs, it didn’t improve our performance that much. So, we used 10 epochs. In the case of the activation function, we have used ‘ReLU’ and we used ‘Softmax’ as a classification algorithm which is also an activation function. Which reduced training loss.

4.3 Confusion Matrix

Detecting sign language gestures has become very popular among the researchers. In order to get the best out of our model, we have trained our model with multiple classes of image and used it as the final layer of our Deep CNN model.

The main anticipating analytics like accuracy, recall, specificity and precision are shown applying confusion matrices. Confusion matrices are important because it let you analyze values like positives that are true, positives that aren't true, negatives that are true, and negatives that aren't true in a simple way. In order to evaluate the performance of our model, we have used confusion matrices. Here in Figure 6, we have shown the confusion matrix of the Deep CNN model after training the model with our given images. The diagonal part indicates the number of images it has been detected correctly.

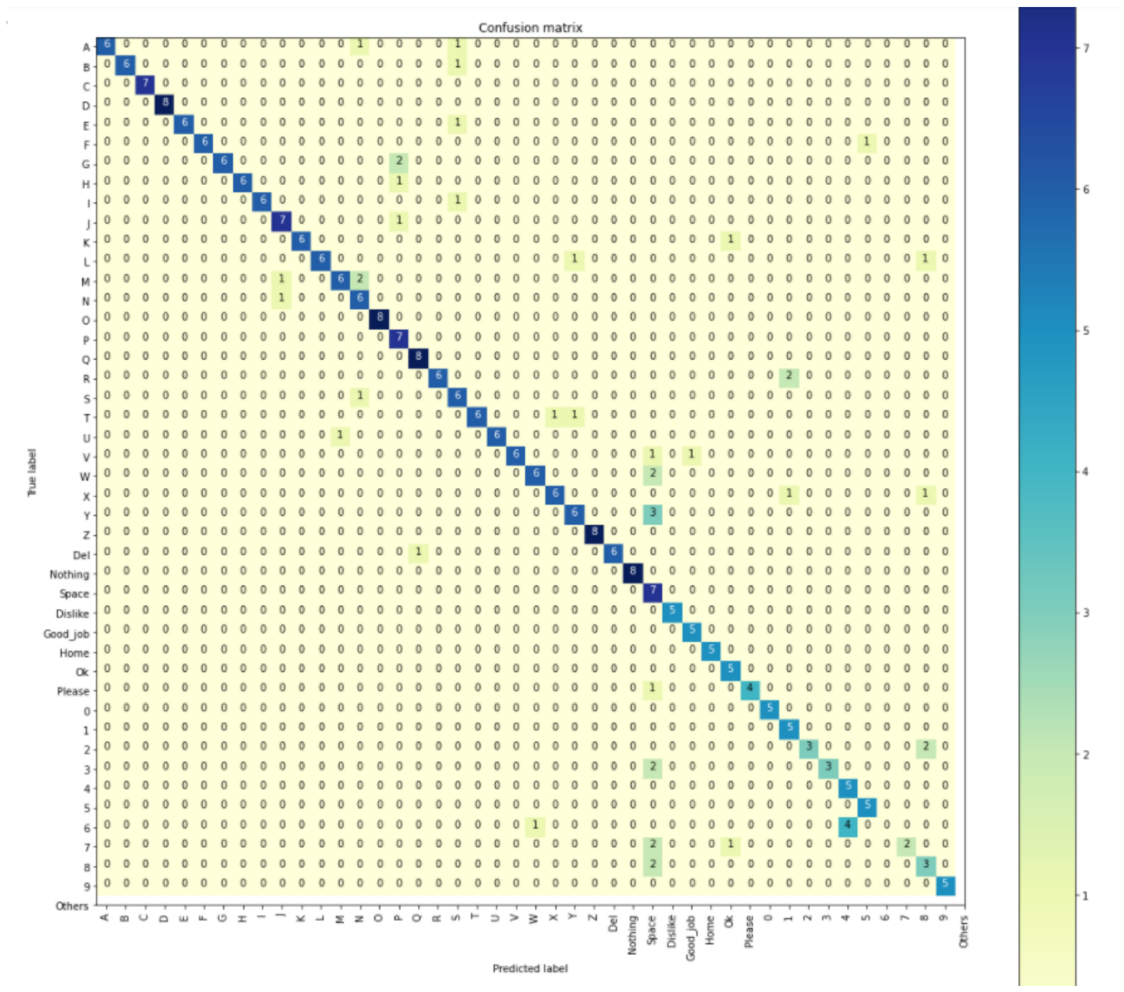

Figure 6: Confusion matrix of Deep CNN

Here in Figure 7, we have shown the confusion matrix of the VGG16 model after training the model with our given images. The diagonal part indicates the number of images it has been detected correctly.

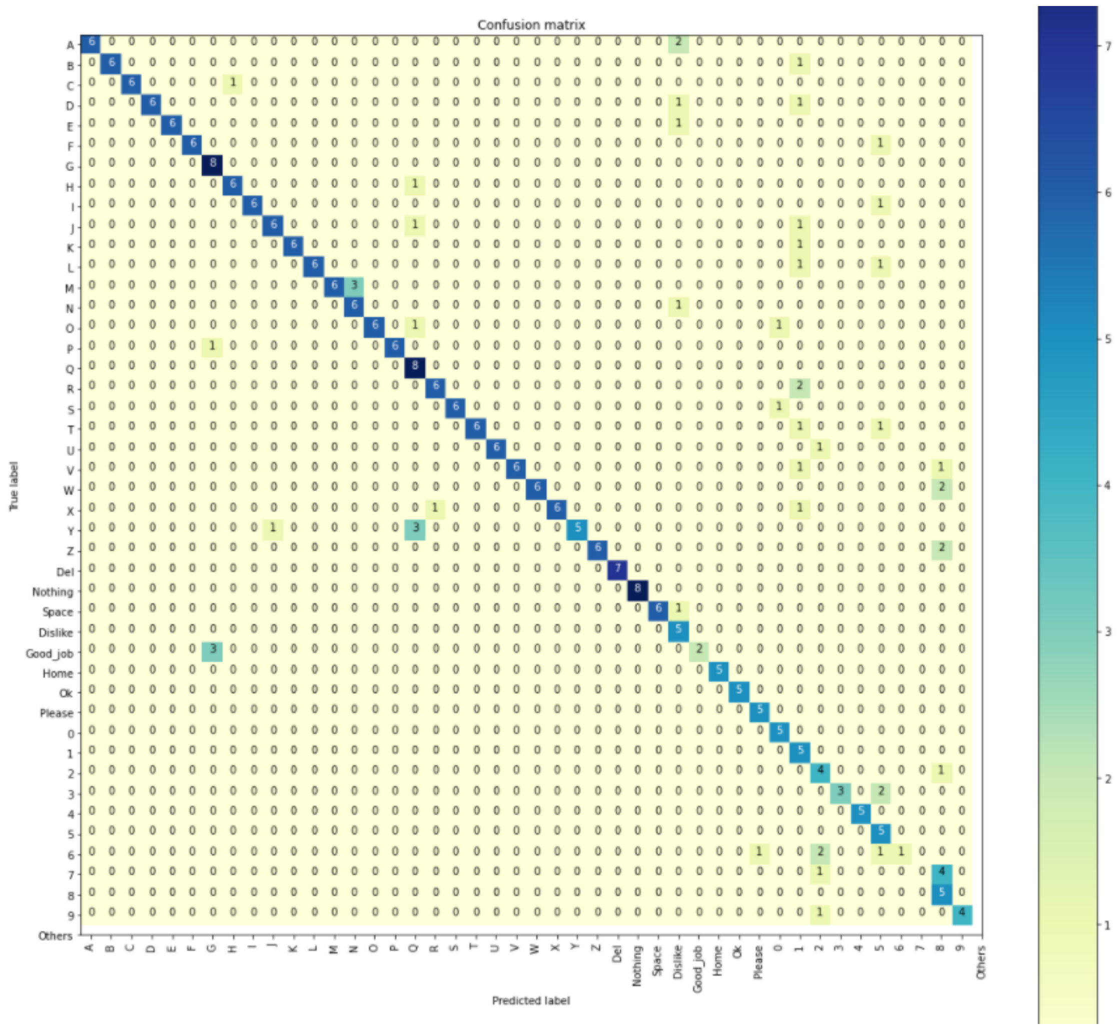

Figure 7: Confusion matrix of VGG16

Here in Figure 8, we have shown the confusion matrix of our improved Deep CNN model after training the model with our given images. The diagonal part indicates the number of images it has been detected correctly.

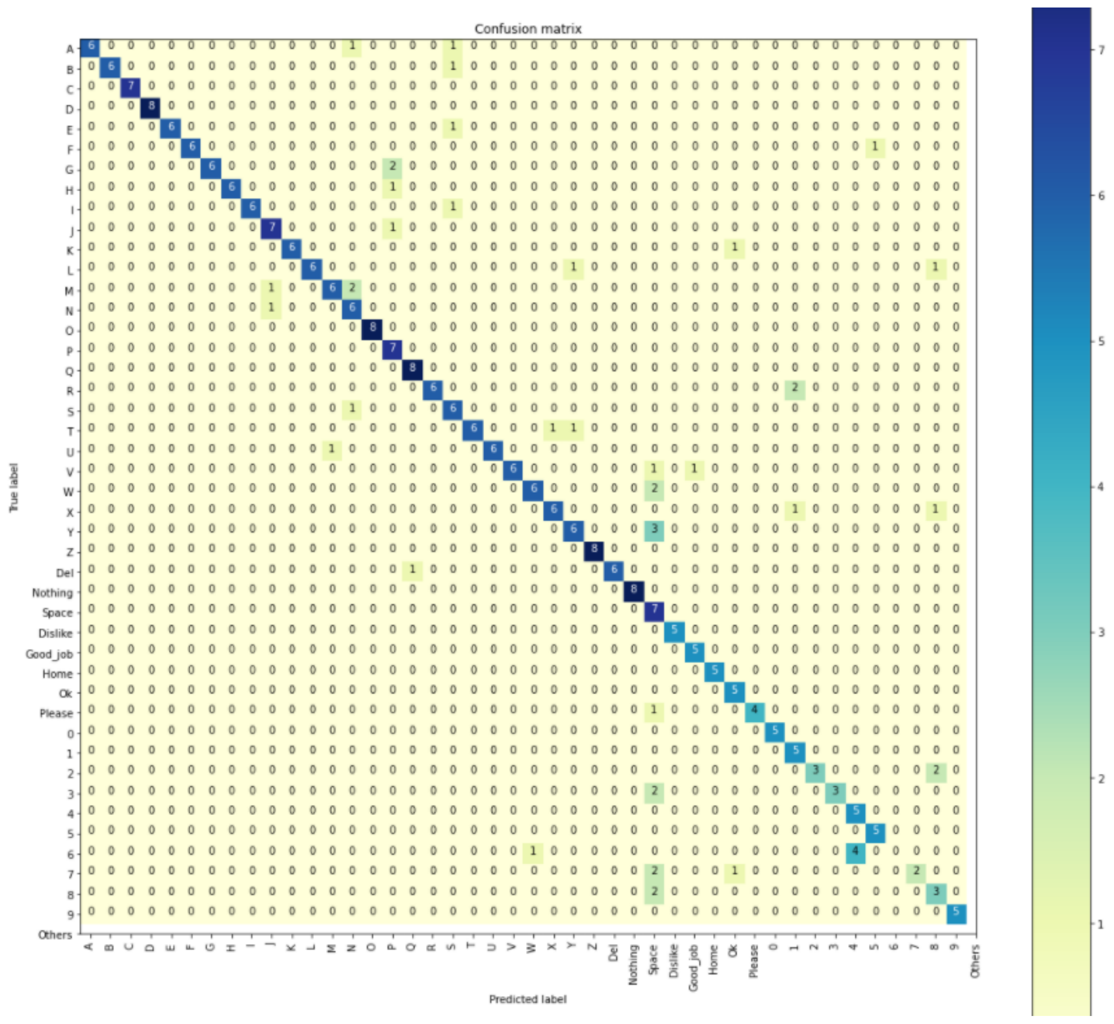

Figure 8: Confusion matrix of Improved Deep CNN

4.4 Performance Analysis

Here, our proposed transfer learning-based Deep CNN model outperformed the general DCNN model and the pre-trained VGG16 model. This Deep CNN model is easy to implement in Tensorflow and Keras framework. It also can detect features from images automatically which makes it so easy to use. If we analysis the 3 different models we used in our project as we can see, the performance of the transfer learning-based Deep CNN model is very good. For multiclass image classification, Deep CNN is the best model. It also takes less time to run the procedure which is very important for a model like this.

The table we have created here is basically one of the different models we have tested on our 45 classes. Here we measure precision, recall, f1-score and support. Precision, also common as a high probability of success, is the proportion of incidences that are applicable among the recovered example, whereas recall, also known as sensitivity, is the fraction of retrieved relevant instances. Precision and recall are thus determined by relevance. On the other hand the harmonic mean of accuracy and recall is given by the f1-score. The scores assigned to each class indicate the classifier's accuracy in categorizing data points in that class when compared to all other classes. The number of samples of the real answer that fall into that class constitutes the support.

From this table we understand very quickly which model is better than other model by measuring all model precision, recall, f1-score, and support.

Table 4: Comparison of Precision, recall and f1-score for Deep CNN, VGG16 and improved Deep CNN model.

Algorithm	**Class**	**Precision**	**Recall**	**F1-score**
Deep CNN	A	1.00	0.75	0.86
VGG16		1.00	0.75	0.86
Improved DCNN		1.00	0.75	0.86
Deep CNN	B	1.00	0.86	0.92
VGG16		1.00	0.86	0.92
Improved DCNN		1.00	0.86	0.92
Deep CNN	C	1.00	1.00	1.00
VGG16		1.00	0.86	0.92
Improved DCNN		1.00	1.00	1.00
Deep CNN	D	1.00	1.00	1.00
VGG16		1.00	0.75	0.86
Improved DCNN		1.00	1.00	1.00

Algorithm	Class	Precision	Recall	F1-score
Deep CNN	E	1.00	0.86	0.92
VGG16		1.00	0.86	0.92
Improved DCNN		1.00	0.86	0.92
Deep CNN	F	1.00	0.86	0.92
VGG16		1.00	0.86	0.92
Improved DCNN		1.00	0.86	0.92
Deep CNN	G	1.00	0.75	0.86
VGG16		0.67	1.00	0.80
Improved DCNN		1.00	0.75	0.86
Deep CNN	H	1.00	0.86	0.92
VGG16		0.86	0.86	0.86
Improved DCNN		1.00	0.86	0.92
Deep CNN	I	1.00	0.86	0.92
VGG16		1.00	0.86	0.92
Improved DCNN		1.00	0.86	0.92
Deep CNN	J	0.78	0.88	0.82
VGG16		0.86	0.75	0.80
Improved DCNN		0.78	0.88	0.82
Deep CNN	K	1.00	0.86	0.92
VGG16		1.00	0.86	0.92
Improved DCNN		1.00	0.86	0.92
Deep CNN	L	1.00	0.75	0.86
VGG16		1.00	0.75	0.86
Improved DCNN		1.00	0.75	0.86

Algorithm	**Class**	**Precision**	**Recall**	**F1-score**
Deep CNN	M	0.86	0.67	0.75
VGG16		1.00	0.67	0.80
Improved DCNN		0.86	0.67	0.75
Deep CNN	N	0.60	0.86	0.71
VGG16		0.67	0.86	0.75
Improved DCNN		0.60	0.86	0.71
Deep CNN	O	1.00	1.00	1.00
VGG16		1.00	0.75	0.86
Improved DCNN		1.00	1.00	1.00
Deep CNN	P	0.64	1.00	0.78
VGG16		1.00	0.86	0.92
Improved DCNN		0.64	1.00	0.78
Deep CNN	Q	0.89	1.00	0.94
VGG16		0.57	1.00	0.73
Improved DCNN		0.89	1.00	0.94
Deep CNN	R	1.00	0.75	0.86
VGG16		0.86	0.75	0.80
Improved DCNN		1.00	0.75	0.86
Deep CNN	S	0.60	0.86	0.71
VGG16		1.00	0.86	0.92
Improved DCNN		0.60	0.86	0.71
Deep CNN	T	1.00	0.75	0.86
VGG16		1.00	0.75	0.86
Improved DCNN		1.00	0.75	0.86
Deep CNN	U	1.00	0.86	0.92
VGG16		1.00	0.86	0.92
Improved DCNN		1.00	0.86	0.92

Algorithm	Class	Precision	Recall	F1-score
Deep CNN	V	1.00	0.75	0.86
VGG16		1.00	0.75	0.86
Improved DCNN		1.00	0.75	0.86
Deep CNN	W	0.86	0.75	0.80
VGG16		1.00	0.75	0.86
Improved DCNN		0.86	0.75	0.80
Deep CNN	X	0.86	0.75	0.80
VGG16		1.00	0.75	0.86
Improved DCNN		0.86	0.75	0.80
Deep CNN	Y	0.75	0.67	0.71
VGG16		1.00	0.56	0.71
Improved DCNN		0.75	0.67	0.71
Deep CNN	Z	1.00	1.00	1.00
VGG16		1.00	0.75	0.86
Improved DCNN		1.00	1.00	1.00
Deep CNN	0	1.00	1.00	1.00
VGG16		0.71	1.00	0.83
Improved DCNN		1.00	1.00	1.00
Deep CNN	1	0.62	1.00	0.77
VGG16		0.33	1.00	0.50
Improved DCNN		0.62	1.00	0.77
Deep CNN	2	1.00	0.60	0.75
VGG16		0.44	0.80	0.57
Improved DCNN		1.00	0.60	0.75
Deep CNN	3	1.00	0.60	0.75
VGG16		1.00	0.60	0.75
Improved DCNN		1.00	0.60	0.75

Algorithm	**Class**	**Precision**	**Recall**	**F1-score**
Deep CNN	4	0.56	1.00	0.71
VGG16		1.00	1.00	1.00
Improved DCNN		0.56	1.00	0.71
Deep CNN	5	0.83	1.00	0.91
VGG16		0.42	1.00	0.59
Improved DCNN		0.83	1.00	0.91
Deep CNN	6	0.00	0.00	0.00
VGG16		1.00	0.20	0.33
Improved DCNN		0.00	0.00	0.00
Deep CNN	7	1.00	0.40	0.57
VGG16		0.00	0.00	0.00
Improved DCNN		1.00	0.40	0.57
Deep CNN	8	0.43	0.60	0.50
VGG16		0.33	1.00	0.50
Improved DCNN		0.43	0.60	0.50
Deep CNN	9	1.00	1.00	1.00
VGG16		1.00	0.80	0.89
Improved DCNN		1.00	1.00	1.00
Deep CNN	Delete	1.00	0.86	0.92
VGG16		1.00	1.00	1.00
Improved DCNN		1.00	0.86	0.92
Deep CNN	Nothing	1.00	1.00	1.00
VGG16		1.00	1.00	1.00
Improved DCNN		1.00	1.00	1.00
Deep CNN	Dislike	1.00	1.00	1.00
VGG16		0.45	1.00	0.62
Improved DCNN		1.00	1.00	1.00

Algorithm	**Class**	**Precision**	**Recall**	**F1-score**
Deep CNN	Good job	0.83	1.00	0.91
VGG16		1.00	0.40	0.57
Improved DCNN		0.83	1.00	0.91
Deep CNN	Home	1.00	1.00	1.00
VGG16		1.00	1.00	1.00
Improved DCNN		1.00	1.00	1.00
Deep CNN	OK	0.71	1.00	0.83
VGG16		1.00	1.00	1.00
Improved DCNN		0.71	1.00	0.83
Deep CNN	Please	1.00	0.80	0.89
VGG16		0.83	1.00	0.91
Improved DCNN		1.00	0.80	0.89
Deep CNN	Space	0.35	1.00	0.52
VGG16		1.00	0.86	0.92
Improved DCNN		0.35	1.00	0.52
Deep CNN	Others	0.00	0.00	0.00
VGG16		0.00	0.00	0.00
Improved DCNN		0.00	0.00	0.00

Here we show the pixel intensity histogram of an image from out training dataset.

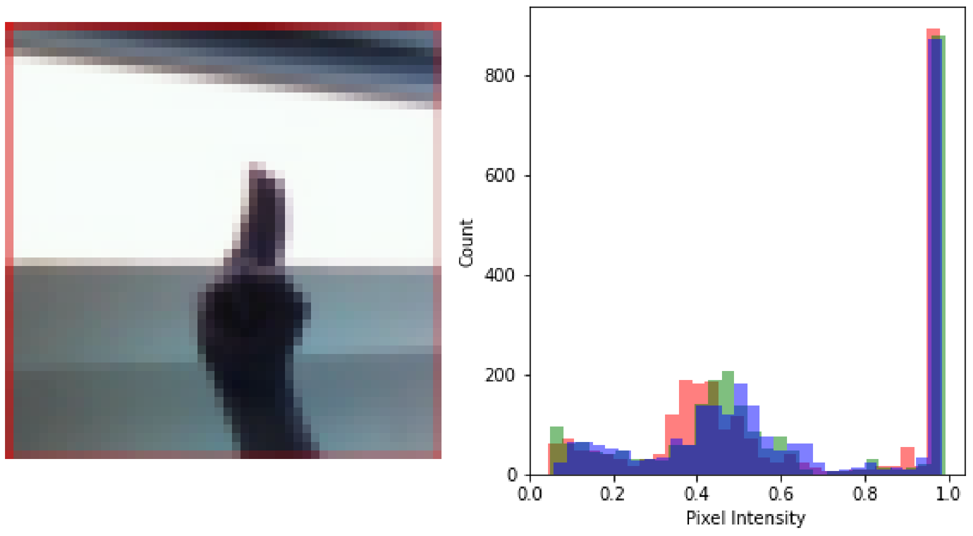

Figure 9: Pixel intensity histogram

Here we have shown a sign language which represents alphabet “E” and it is correctly defined.

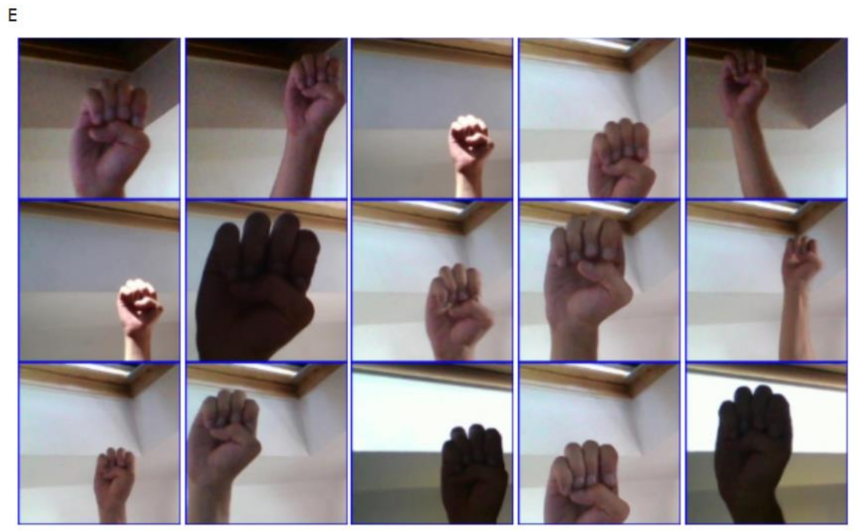

Figure 10: Character representation of “E”

On the test data, 93.57% accuracy was attained using the suggested model. In the relevant field, it improves by 2.43% over the previous high of 95%. The suggested model has a validation loss of 8.24%.The model has been created and trained on the arranged dataset in this manner that it provides almost accurate results. The performance graphs give a clearer picture of the outcomes. The vertical and horizontal axes have been rescaled by 1/100 and 1/10, respectively. The horizontal axis indicates the number of epochs, while the vertical axis is the percent accuracy or loss. The suggested model's performance accuracy curve is shown in Figure 11.

The proposed model's highest validation accuracy has been determined to be 99.57%. For increasing epochs, no additional suggestive improvement was found for the proposed model.

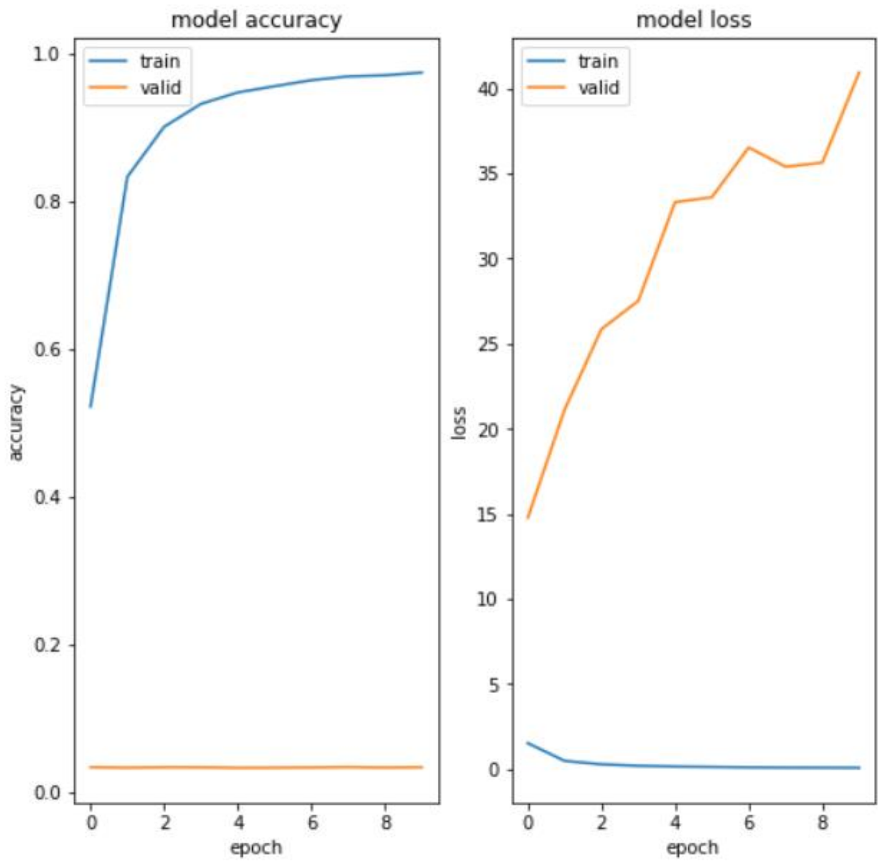

Figure 11: Learning curve of custom CNN model.

The validation loss and accuracy curve of the proposed model is shown in Figure 12. The validation loss has been shown to saturate at 0.56%. A comparison chart showing the accuracies of comparable works provides a realistic picture of performance improvements. The suggested model in this study recognizes all 46 Sign Language characters with high accuracy and detects them in real time. The suggested model was additionally validated using test data derived from the hand sign photos of ten outside signers who were not included in the main dataset.

On the test data provided by outsiders, the proposed model obtains a 99.41% accuracy. For the test data in this scenario, the validation loss for the suggested model is 0.70%.

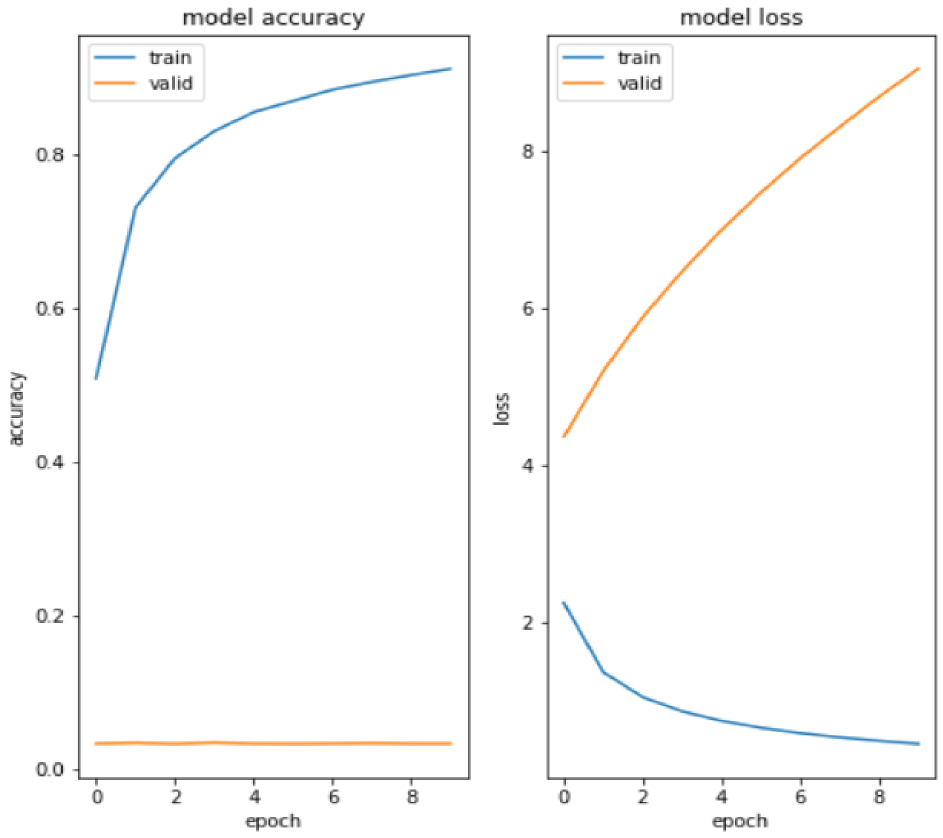

Figure 12: Learning curve of VGG16 model.

Fig 13 show the Improved Deep CNN model's performance accuracy and loss curves in this scenario, the accuracy and loss measurements illustrate the robustness of the proposed model regardless of the signers. The recognition model was used to predict the characters of Bangla Sign Language in various lighting circumstances, and the model was used to extract the shortest transition time between characters. The suggested model was used to predict characters at various angular deviations of certain hand signals, and the related probability distribution of the characters was recovered from the model. Up to 30 degrees of angular variation, the model gives significant predictability for the hand signals of Sign Language characters.

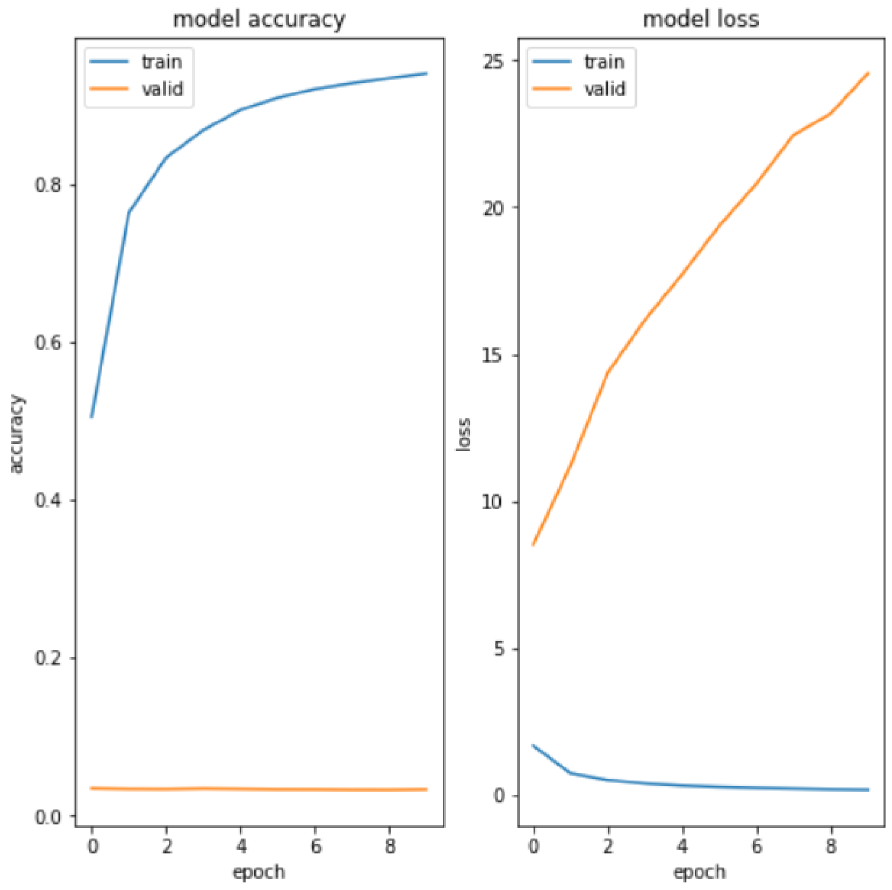

Figure 13: Learning curve of Improved Deep CNN model.

Comparative analysis means comparing our model with other traditional ones. After comparing accuracy with the VGG16 model and a general Deep CNN model, we get Table 5. Here, Table 5 gives detail of result analysis.

Table 5: Performance comparison image detection with some traditional methods

Method Name	**Accuracy**
Deep CNN	94.56%
VGG16	92.88%
Improved Deep CNN	95.02%

We have also used two more datasets form kaggle and trained the datasets into the same model we created. After analyzing the three models we can say that the Improved Deep CNN method gives better performance than other methods. Here, Table 6 gives detail of result analysis of the three models for Improved Deep CNN method.

Table 6: Performance comparison of the proposed model using different Datasets

Dataset	**Method Name**	**Accuracy**
ASL Alphabet by Akash [18]	Improved Deep CNN	95.02%
Sign Language Gesture Images Dataset by Ahmed Khan [19]		98.65%
American Sign Language Dataset by Ayush Thakur [20]		99.51%

CHAPTER 5
CONCLUSION AND FUTURE WORK

This research study proposed a hand gesture detection and sign language recognition system using Deep CNN with VGG16 of ImageNet. In terms of detecting hand gestures, it gave good results with Deep CNN and VGG16. When we used transfer learning our Improved Deep CNN model gave better results than the other methods. Moreover, it gave 95% accuracy over all the classes. Here we have used image data of many hand gestures. Which we hope will be helpful for the speech and hearing-impaired peoples in the future if we develop this project. In our current model, we cannot detect hand gestures from a real time data or video data. In the future, we will build a model which will be able to detect the hand gestures from a real time or video data. Also, we will increase the variations of the hand gestures and more words so that it can detect more words from different angles.

References

[1] Shivashankara S, Srinath S, " American Sign Language Recognition System: An Optimal Approach ", International Journal of Image, Graphics and Signal Processing(IJIGSP), Vol.10, No.8, pp. 18-30, 2018. DOI: 10.5815/ijigsp.2018.08.03

[2] R. Elakkiya “Machine learning based sign language recognition: a review and its research frontier” Journal of Ambient Intelligence and Humanized Computing https://doi.org/10.1007/s12652-020-02396-y Received: 23 April 2020 / Accepted: 23 July 2020 © Springer-Verlag GmbH Germany, part of Springer Nature 2020.

[3] D. Forsyth, A. Farhadi and R. White, "Transfer Learning in Sign language," in 2007 IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, 2007 pp. 1-8. doi: 10.1109/CVPR.2007.383346

[4] Lean Karlo S. Tolentino, Ronnie O. Serfa Juan, August C. Thio-ac, Maria Abigail B. Pamahoy, Joni Rose R. Forteza, and Xavier Jet O. Garcia “Static Sign Language Recognition Using Deep Learning” International Journal of Machine Learning and Computing, Vol. 9, No. 6, December 2019.

[5] 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., 2011, Vol. 15, pp. 12–20 Available online at http://iims.massey.ac.nz/research/letters/.

[6] Md. Shahinur Alam, Mahib Tanvir, Dip Kumar Saha, Sajal K. Das “Two Dimensional Convolutional Neural Network Approach for Real-Time Bangla Sign Language Characters Recognition and Translation” Received: 16 February 2021 / Accepted: 18 July 2021 The Author(s), under exclusive licence to Springer Nature Singapore Pte Ltd 2021.

[7] Teak-Wei Chong , Boon-Giin Lee “American Sign Language Recognition Using Leap Motion Controller with Machine Learning Approach” Received: 11 September 2018; Accepted: 17 October 2018; Published: 19 October 2018.

[8] Deza, Anna, and Danial Hasan. "MIE324 Final Report: Sign Language Recognition." in 2nd December 2018 .

[9] J. 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, 2016, pp. 86–90.

[10] Cheok, M.J., Omar, Z. & Jaward, M.H. “A review of hand gesture and sign language recognition techniques”. Int. J. Mach. Learn. & Cyber. 10, 131–153 (2019). https://doi.org/10.1007/s13042-017-0705-5

[11] S. Upendran and A. Thamizharasi, "American Sign Language interpreter system for deaf and dumb individuals," 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014, pp. 1477-1481, doi: 10.1109/ICCICCT.2014.6993193.

[12] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convo lutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)

[13] Ivan Gruber, Dmitry Ryumin, Marek Hr´uz, and Alexey Karpov “Sign Language Numeral Gestures Recognition Using Convolutional Neural Network”.

[14] Barbhuiya, A.A., Karsh, R.K. & Jain, R. “CNN based feature extraction and classification for sign language”. Multimed Tools Appl 80, 3051–3069 (2021), Published: 18 September 2020 https://doi.org/10.1007/s11042-020-09829-y

[15] S. J. Pan and Q. Yang, &quot;A Survey on Transfer Learning,&quot; in IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345-1359, Oct. 2010

[16] Lionel Pigou(B), Sander Dieleman, Pieter-Jan Kindermans, and Benjamin Schrauwen “Sign Language Recognition Using Convolutional Neural Networks”.

[17] Abdul Muntakim Rafi, Nowshin Nawal, Nur Sultan Nazar Bayev, Lusain Nima, Dr. Celia Shahnaz, Shaikh Anowarul Fattah "Image-based Bengali Sign Language Alphabet Recognition for Deaf and Dumb Community" 2019 IEEE Global Humanitarian Technology Conference (GHTC).

[18] Akash. ASL Alphabet. url: https://www.kaggle.com/grassknoted/asl-alphabet. (accessed: 14.10.2021).

[19] Ahmed Khan. Sign Language Gesture Images Dataset. url: https://www.kaggle.com/ahmedkhanak1995/sign-language-gesture-images-dataset (accessed: 03.01.2022).

[20] Ayush Thakur. American Sign Language Dataset. url: https://www.kaggle.com/ayuraj/asl-datase (accessed: 03.01.2022).

[21] Shaha, Manali & Pawar, Meenakshi "Transfer Learning for Image Classification". Proceedings of the 2nd International conference on Electronics, Communication and Aerospace Technology (ICECA 2018) IEEE Conference Record # 42487; IEEE Xplore ISBN:978-1-5386-0965-1.

[22] Mahbub Hussain, Jordan J. Bird, and Diego R. Faria "A Study on CNN Transfer Learning for Image Classification". Conference: UKCI 2018: 18th Annual UK Workshop on Computational Intelligence.

Appendix

This project is done in combination with the CSE-499 Project/Internship final project.