Project Report Entitled

"INDIAN SIGN LANGUAGE RECOGNITION USING DEEP LEARNING FRAMEWORKS"

Submitted in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY (B.Tech)

in

INFORMATION TECHNOLOGY

SUBMITTED BY

ESHA GAVALI	**[C No: C22016441509]**
AYESHATASNIM HANNURE	**[C No: C22016771739]**
SHIVANI KAMTIKAR	**[C No: C22016441521]**
URVI LENDHE	**[C No: C22016441528]**

Under the guidance of

Dr. Anagha Kulkarni

DEPARTMENT OF INFORMATION TECHNOLOGY

MKSSS'S CUMMINS COLLEGE OF ENGINEERING FOR WOMEN, PUNE

(An Autonomous Institute Affiliated to Savitribai Phule Pune University)

July 2020

CERTIFICATE

This is to certify that this Project report titled "**INDIAN SIGN LANGUAGE RECOGNITION USING DEEP LEARNING FRAMEWORKS**" submitted by

1. ESHA GAVALI [C No: C22016441509]
2. AYESHATASNIM HANNURE [C No: C22016771739]
3. SHIVANI KAMTIKAR [C No: C22016441521]
4. URVI LENDHE [C No: C22016441528]

is a record of bonafide work carried out by them, under my guidance, in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology (B.Tech) in Information Technologyof MKSSS's Cummins College of Engineering For Women, Pune, affiliated to Savitribai Phule Pune University.

Dr. Anagha Kulkarni
Guide
Dept. of Information Technology

Dr. Anagha Kulkarni
Head.
Dept. of Information Technology

Dr. Madhuri Khambete
Principal
MKSSS's Cummins College of Engineering For Women, Pune

Date:

Place: Pune

REPORT APPROVAL

Project Report Entitled

"INDIAN SIGN LANGUAGE RECOGNITION USING DEEP LEARNING FRAMEWORKS"

By

ESHA GAVALI	**[C No: C22016441509]**
AYESHATASNIM HANNURE	**[C No: C22016771739]**
SHIVANI KAMTIKAR	**[C No: C22016441521]**
URVI LENDHE	**[C No: C22016441528]**

is approved for the degree of Bachelor of Technology

of

DEPARTMENT OF INFORMATION TECHNOLOGY

MKSSS'S CUMMINS COLLEGE OF ENGINEERING FOR WOMEN, PUNE

(An Autonomous Institute Affiliated to Savitribai Phule Pune University)

Dr. Anagha Kulkarni
Guide

External Examiner

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke action from the sources which have thus been properly cited or from whom proper permission has not been taken when needed.

Esha Gavali
Ayeshatasnim Hannure
Shivani Kamtikar
Urvi Lendhe

Date:

ACKNOWLEDGEMENTS

Firstly, we would like to thank the director Dr. Madhuri Khambete and our institution, MKSSS's Cummins College of Engineering for providing us with an opportunity to implement our ideas. We would like to express our sincere gratitude to Dr. Anagha Kulkarni for her continual guidance and encouragement during the course of the project. We would also like to thank our reviewers Mrs. Madhura Tokekar and Mrs. Radhika Bhagwat for their advice and feedback. The inputs from both, our guide and our reviewers helped us refine and structure the project, and helped put it into the form it is now. We also want to acknowledge the efforts of our families, friends, classmates and Mr. Mangamuri Leela Surya Teja who contributed to the creation of the dataset. A final token of appreciation for each team member for their persistent dedication, and constant cooperation.

Esha Gavali
Ayeshatasnim Hannure
Shivani Kamtikar
Urvi Lendhe

ABSTRACT

The Indian Sign Language (ISL) is a language used for communication by over several hundred deaf and verbally challenged people in the Indian Subcontinent. It is a relatively newly defined language by the Indian Sign Language Research and Training Centre (ISLRTC) and the 2nd Edition of ISL Dictionary was released as recently as the 27th of February, 2019. Owing to novelty of ISL, not a lot of people are aware of it and many verbally impaired individuals have not yet adopted ISL. There is a significant communication gap between the verbally impaired and the rest of the society and the aim of the project is to bridge this gap by providing a means to identify and recognise the gestures signed by the deaf community so as to facilitate smooth communication. This project is especially challenging since no legitimate dataset is available and hence no previous work had been done on ISL at the time of inception of our project. This project's preliminary goal was to generate a novel dataset using the official ISL dictionary as a reference. In order to make our model robust, we incorporated images varying in background, colour, orientation and lighting in the dataset. Next, we compared the accuracy and working of various deep learning models and with the help of this analysis, created our own Convolutional Neural Network (CNN) model from scratch. We experimented by changing different parameters in the architecture of the model such as, epoch size, pooling methods, activation functions and loss functions. Upon observing and studying the results of these experiments, an architecture was finalised for our CNN model which was best suited for our application and data. This model is evaluated using various metrics such as precision, recall, f1 score etc.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

ASL	American Sign Language
IP	Image Processing
ISL	Indian Sign Language
ISLRTC	Indian Sign Language Research and Training Centre
CNN	Convolutional Neural Network
FCN	Fully Connected Network
ReLU	Rectified Linear Unit
VGG	Visual Geometry Group
NN	Neural Network
ROC	Receiver Operating Characteristic
PIL	Python Imagine Library
RNN	Recurrent Neural Network

LIST OF SYMBOLS

$F(x, W_i)$	Residual mapping
x	Input vector
y	Output vector
W_s	Linear Projection

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Indo-Pakistani Sign Language (IPSL) or more commonly known as Indian Sign Language (ISL) is the primary sign language used in South Asia by a very few hundred thousand deaf and other verbally challenged people. Unlike American Sign Language (ASL), ISL is in it's rudimentary stages of development. In the 2000s, the Indian hearing-impaired community gave a public recommendation for an institution that focused on ISL teaching and research. The 11th Five Year Plan (2007-2012) acknowledged that the needs of people with hearing disabilities had been relatively disregarded. It anticipated the development of a sign language research and training center, to promote and develop sign language and training of teachers and interpreters. Setting up of the Indian Sign Language Research and Training Centre (ISLRTC) was declared by the Finance Minister in the Union Budget speech of 2010-11. ISLRTC had released the 1st Edition of the Indian Sign Language (ISL) dictionary consisting of 3000 words on 23rd March 2018. After the release of the 2nd Edition, the ISL now consists of 6,000 words commonly used in day-to-day life. ISL in reality is not related to any language such as English or Hindi. It is a common assumption that it is the manual representation of those two languages. ISL has its own grammar. Many linguists discuss three aspects of ISL: its lexicon, syntax and spatial grammar. Following are the main features of ISL that makes it different from other sign languages:

1) Number Signs: To make signs of numbers from zero to nine, you need to hold up the hand/hands with appropriate hand gestures and shapes. The numbers one, two, three, four and five are indicated by the number of extended fingers. For zero and the numbers from six to nine, there are special hand shapes used which are derived from written numbers.

2) Alphabet signs: Signs for alphabets can be made in similar manner, i.e. by making appropriate shapes of the hands. Some alphabets require just one hand and

some require both the hands. There are selective alphabets which require dynamic gestures to form those particular letters.

3) Family Relationship: To make signs for family relationships, the signs are preceded by the sign for male(or man) and female(or woman).

According to 2011 Census, the total population of verbally impaired in India numbered to about 50 lac. Ignorance of the basic needs of this community still makes them be a part of one of the weakest sections of the society and difficulty in their lifestyles. Problems of the deaf community have been documented by various organizations working for the deaf. Obsolete training methods and teaching systems pertaining to this section of the society require urgent attention.

1.2 RELEVANCE

ISL is used by the deaf community across India. However, most of the deaf schools do not teach hearing impaired children using ISL. Teachers teaching in such schools are not are not oriented towards teaching methods that use ISL during their training programs. Moreover, there is a scarcity of teaching material that incorporates sign language. Additionally, parents of deaf children (specially in rural India) are not aware about sign language and do not avail its ability to remove communication barriers. Institutes and places where communication between deaf and hearing people takes place, ISL interpreters are an urgent requirement. However, India has less than 300 certified interpreters, which are insufficient to meet the demand. Deaf schools in many regions in the Indian subcontinent are overwhelmingly oral in their approach. Hence, owing to the lack of resources in the educational sector, verbally challenged individuals are forced to drop out of school and abstain from pursuing higher education.

The Deaf communities of India are still facing issues while trying to get ISL to gain the status of a minority language. Our goal is to remove the ignorance pertaining to this subject and the help to normalise ISL and help to facilitate smooth and unhindered communication between all parts of society. Building a software that not only recognises the gestures signed by the deaf community but also translates it into English would help the user understand and communicate with the deaf community better.

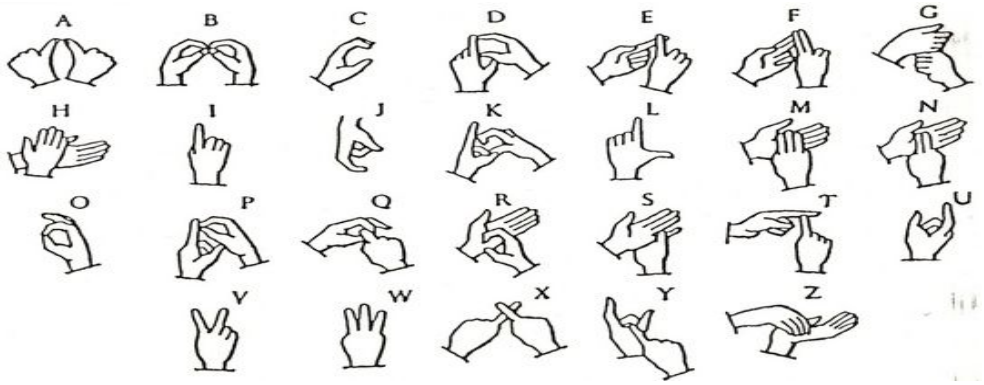


Figure 1.1: Alphabets in ISL

1.3 PROJECT UNDERTAKEN

Considering that the deaf community is underrepresented and ISL is still in the rudimentary stages, our team recognised the need to make society aware of ISL and facilitate a smooth communication between all sections of society. The objective of our project is to build a software that recognises the gestures signed by the verbally impaired and convert it to the corresponding English language alphabet, number, word, phrase or sentence. In our project, we filtered the static gestures in the ISL dictionary and have worked on recognising these gestures. The biggest challenge faced by us during the project was the unavailability of a cohesive dataset and hence, our project started right from creating our own ISL dataset from scratch to building a full fledged deep learning model that recognises the gestures. Since ISL is a comparatively new field, it has not been explored extensively whereas abundant projects, research, studies etc. has been done on American Sign Language (ASL) and other famous sign language systems. Taking inspiration from projects and research done on other sign language systems, we decided to build our own model that would recognise gestures in Indian Sign Language. We wish to gain more knowledge on the Indian Sign Language and its complexities by understanding the nuances in ISL and other sign language systems. In this project, we will be using various programming, image pre-processing and deep learning constructs which would reinforce our knowledge and expertise on the subject.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing Sign Language Recognition Systems

Since the official dictionary of ISL was recently released, this area is comparatively unexplored with ISL being in is in its rudimentary stages of development. After the release of the second edition, the ISL consists of 6,000 words commonly used in day-to-day life. ISL has been comparatively unexplored. There are various dialects of ISL depending on the region, culture and history of the signer and it has its own grammar. Therefore, standardisation of ISL has not taken place. As opposed to this, ASL and other popular sign language systems have been extensively worked upon. The research papers that are referenced are discussed below in short:

[1] This is a research paper that studies real-time Sign Language Recognition System for South Indian Language. Here, authors have used a total of 32 signs. Image processing techniques were used on static images and using the position of finger-tips, the alphabets were recognised and converted into text. The method proposed allows identification of the images captured dynamically during testing. Test data results show that the proposed Sign Language Recognition System can recognise input signs with an accuracy of 98.125%. And the train and test image count was 320 and 160 respectively.

[2] An automatic gesture recognition approach for Indian Sign Language is proposed by this research. Both the hands were used in signage of every alphabet. The authors addressed identification of local-global ambiguity along with enhancement of inter-class variability in each hand gesture in their approach. To classify each hand gesture, multi class non linear support vector machines (SVM) is used which resulted in rate of alphabet recognition being 91.3%. In this study, the derived alphabets were from both the British Sign Language and the French Sign Language.

The paper on [3], based on ASL, mentions the importance of finger-spelling and how translation of alphabets should be given more priority than words. The paper studies the usage of Deep Convolutional Networks for Gesture Recognition in the ASL. The

approach was using a mini-batch stochastic gradient descent known to be associated with a fundamental supervised learning algorithm. The authors use two datasets to see the results: a pre-made one and a manually generated one. The results on pre-made datasets are accurate as the dataset is uniform, however, the results on a manually generated dataset are not very accurate as the data is collected under normal settings and a regular camera.

The paper [4], incorporated the Neural Network (NN) with Genetic Algorithm (GA), Evolutionary algorithm (EA) and Particle Swarm Optimization (PSO) separately to attain three singular methods namely NN-GA, NN-EA and NN-PSO respectively, as ways to effectively recognize ISL gestures.

The authors of [5], propose a system to recognise ISL in real-time. They use flipping as a data augmentation technique. Transfer learning with a CNN model called GoogLeNet for classification is used. The experiments are performed for recognizing alphabets (excluding j and z).

2.2 Deficiencies of the Existing Art

The drawbacks of earlier works discussed above include: Many techniques have been proposed to translate ISL alphabets and numbers to English by using gloves, a set of multiple cameras, Kinect camera that gives a 3-D view and sensor gloves to capture sensor values to detect hand gestures. These methods are costly and have setup requirements. There is a need to identify the base of the palm first and then the fingertips. If these steps are not carried out accurately, then final results might be affected. Software based techniques to achieve the task is easier. Variety of methods have proved to be better than the methods using special devices. Multi-class SVM involves a tedious task of creating multiple one-vs-rest models. Moreover, SVM does not work for dynamic images. Images have to be converted from their 2-D format to 1-D when using ANN. As a result, the number of trainable parameters increases and the spatial features of the image are lost. The use of CNN is promising in image recognition. Variety of advanced CNN techniques have been proposed for recognising ASL and other western languages. They need to be used for Indian sign language recognition. Based on the literature survey, traditional machine learning algorithms have been observed to have

significant problems. Their biggest drawback is that they tend to get trapped into local optima while optimizing an objective. Moreover, their dependence on the methods of pre-processing used creates more room for error. They also consume large amounts of memory and time. While solving an image classification problem using Artificial Neural Networks, prior to training the model, the 2-dimensional image has to be converted to a 1-dimensional vector. The drawback is that with the increase in the size of the image, the number of trainable parameters increases drastically.

2.3 MODULES

The problem statement undertaken is divided into the following 2 modules:

1. Dataset Construction
   (a) Dataset Generation
   (b) Image Pre-Processing
2. Model Creation

The literature survey is also divided in a similar manner focusing on each module as a separate research entity.

2.3.1 Dataset Construction

DATASET GENERATION

Dataset is a collection of elements and information generated for research studies. Dataset can either be created, i.e generated or can be downloaded, provided by any agencies or non-profit organizations. We can identify data by looking for organizations that focus on our area of interest. For example, in our case, if we want to look for datasets related to ISL we will search for organizations or people who have already generated their data, or we can create our dataset from scratch by such references. Datasets could be of various types such as text, numerical, categorical, images, etc. We will be using image dataset for our project.

Abundant resources are available in the case of ASL and other popular sign languages. However, there is no consistent dataset in the case of ISL. A few datasets that claim to be of ISL are not consistent with the official dictionary of the language.

IMAGE PRE-PROCESSING

Image Processing basically is used for :

- improving images for human viewing
- processing image datasets undergoing some machine algorithms

Image Processing (IP) can be analogue or digital. Digital Image Processing refers to the usage of machine algorithms to process digital images.

Pre-processing

Pre-processing is done so that the raw data images are converted into a format which can support, in this case, deep learning algorithms. A better structured dataset is essential to increase model accuracy. [6]

Pre-processing has three major steps:

1. Data formatting: standardises the datasets
2. Data cleaning: removes noise, cropping
3. Data sampling: uses smaller data to reduce computational power

Feature Extraction and Recognition

Extracting the important features or parts of an image by reducing the dimensionality without compromising on the content is what feature extraction is all about. These important features extracted are then distinguished and labelled into classes for modelling and interpretation. Such kind of data is called tagged or labelled data.

Image Augmentation

Augmenting refers to using one or more processing techniques to artificially create new images which is done to boost a small dataset. This is especially useful when dealing with new projects where a lot of data is not available per class. This also helps to improve accuracy and reduce overfitting. Some major augmentation techniques are:

- Flipping the image
- Rotating the image by slight or huge angles
- Change saturation contrast values

- Convert to Greyscale or change the RGB values

It is very important to choose the correct augmentation techniques for your project so as to not lose important classifying information or render the image completely different and unusable. For this project, we know that we cannot flip, or use large-angle rotation as it will completely change the meaning of the sign.

Image Representation

Image representation is an essential part of Digital Image Processing. It is how you represent every pixel of an image u sing a matrix of rows (x) and columns (y). Hence every image can be written as a 2-dimensional function (f(x,y)) where the values of x and y give us the exact location of any pixel within the image, with the depth or hue ranging from values of 0 to 255 as shown in Figure 2.1.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & f(1,2) & \dots & f(1,N-1) \\ \vdots & \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & f(M-1,2) & \dots & f(M-1,N-1) \end{bmatrix}$$

Figure 2.1: Image Matrix to Locate every Pixel

This representation is then stored as a csv file and fed to the models for classification and recognition.

Comparative Study of Image Processing Methods

In the Hand Gesture Recognition System using Image Processing with Conversion to Text and Speech [7], all the parts of IP have been explained along with the concept of Image Threshholding to achieve segmentation. However, this method has been used for basic hand gestures like peace and victory, and detection requires a dark coloured glove to be worn. Methods like Camshift and Background Subtraction algorithms have been discussed.

Table 2.1: Feature Extraction Accuracy

Feature Extraction Method	Input	Accuracy
combines K curvature and convex hull algorithms	images	94.32%
Color-Based Segmentation, Smoothing	Images from Smart Phone	93%
Gaussian Smoothing, Canny Edge Detector	Images	90.11%
Hand Segmentation and Tracking algorithm	Isolated Images	90%

The Review on Feature Extraction methods of Image based Sign Language Recognition system [8] gives valuable insight on the comparative study of feature extraction methods used on various Image-based Hand Gesture Recognition systems. The most accurate methods were for static images are given in Table 2.1. Interestingly, none of these were used for the ISL. The purpose and the importance of IP and the need for the independency between background, rotation, scale, angle, shape and feature extraction is mentioned.

The paper on Using Deep Convolutional Networks for Gesture Recognition in American Sign Language [9] mentioned the importance of fingerspelling and how translation of alphabets should be given more priority than words. The paper studied the usage of Deep Convolutional Networks for Gesture Recognition in the ASL and they used two datasets to see the results: a pre-made one and a manually generated one. The manually trained dataset was preprocessed by background-subtraction, and was additionally augmented (rotated and flipping), but this dataset gave a much lower accuracy (67%) than the premade dataset (82.5%) due to skin colour and lighting variations, which could be solved by using Microsoft's Kinect Camera.

2.3.2 Existing Deep Learning Models

Convolutional neural Network (CNN)

CNN are the type of Neural Network which are commonly applied to analyze visual imagery. It is similar to a Fully Connected Neural Network with an extra operation added at each layer. A convolution is a mathematical function used to bring down the size of the image without losing any features. Images of large sizes that take a long time to be processed by Neural Networks can be handled using CNN.

Some important concepts related to CNN:

1. **Stride**- The number of pixels the convolution filter shifts.
2. **Padding**- Extra layers of pixels added to each side of the image to maintain a certain size of the image. It also helps include the edge pixels of the image since they get used much less than the centre pixels when processed without padding.
3. **Valid Convolution**- Type of convolution operation that does not have any padding on the input.
4. **Same Convolution**- Type of convolution that adds a padding on the input such that the size of the input and output remains same.

A CNN generally has 3 types of layers:

- **Convolution Layer**- This Layer involves performing the mathematical operation called Convolution.

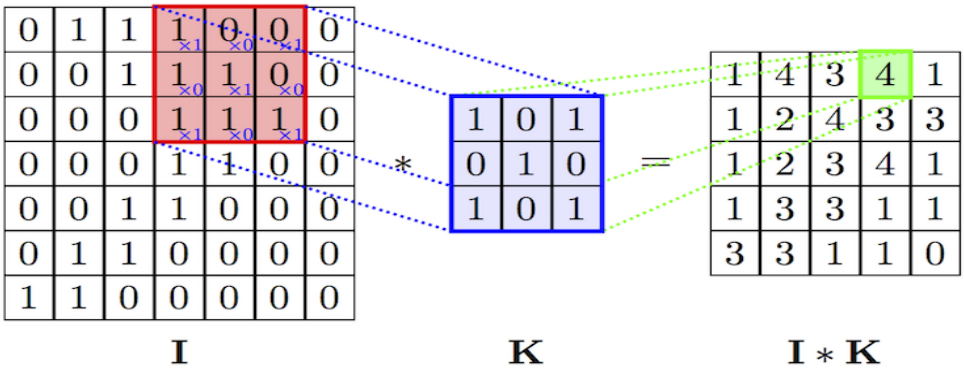


Figure 2.2: Convolution Function

- **Pooling Layer**- This layer does not have any parameters that need to be learnt. Spatial dimensions of the image are reduced without affecting the depth. It helps in improving the performance of the CNN and reduces the chances of overfitting.

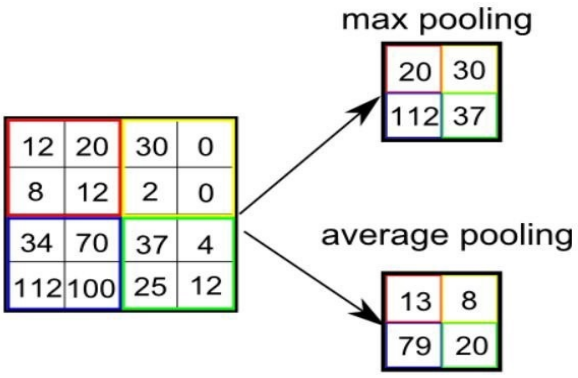


Figure 2.3: Max Pooling and Average Pooling

- **Fully Connected Layer**- This layer is a normal Neural Network layer which has input and output features.

Transfer Learning is a method by which models which are trained on larger, or more general data sets can be used to fit to a specific small data set. A portion of weights are recycled from the pre-trained model instead of training the model by ourselves which may be computationally very expensive and may take days to train.

It is suitable to use transfer learning when-

1. If you are using a specific Neural Network (NN) architecture that has been trained before, you can use this pre-trained parameters/weights instead of random initialization to solve your problem.
2. It can help you boost the performance of the NN
3. The pre-trained models might have trained on a large data sets and took a lot of time to learn those parameters/weights with optimized hyper parameters. This can save you a lot of time.

Visual Geometry Group (VGG) Neural Network

VGG is an example of CNN that was developed for large scale image classification [10]. It takes a 224×224 RGB image as input. It is a deep CNN as it can have up to 19 layers. All the convolutions are of size 3×3 and use Rectified Linear Unit (ReLU) activation functions. It consists 5 maxpool layers of size 2×2 and a stride of 2. It also has 3 Fully Connected Network (FCN), first 2 having 4069 channels and the last one having 1000(number of outputs for the dataset they were working on). The last layer has softmax activation function.

ResNet

Residual networks, commonly called as ResNets have achieved state-of-the-art on challenging computer vision tasks quite recently.[11] Like highway networks, resnets utilize identity shortcut connections associations that empower stream of data over various layers without enervation that would be brought about by numerous stacked non-direct changes, coming about in improved optimization. As stated by the universal approximation theorem, a single layer in a feedforward network is sufficient to represent any function given enough capacity. However, the single layer might be huge and would make the network prone to overfit the data. Therefore, it is a common trend to make the network architecture deeper. However, deepening the network does not simply work by stacking up layer after layer. However, this gives rise to the vanishing gradient problem

- repeated multiplication may make the gradient infinitely small as we back-propagate the gradient to earlier layers. This is due to the multiplication of a number between 0 and 1 many times making it increasingly smaller: thus the gradient begins to “disappear” when reaching the earlier layers. This implies that the earlier layers are slower to train prone to error. This poses a big problem given that the earliest layers are the skeleton of the whole network - they identify the basic core features of the entire network.

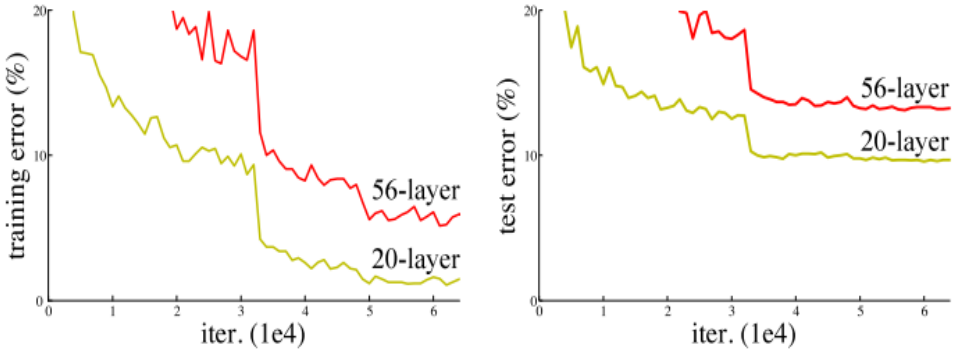


Figure 2.4: Training error (left) and test error

As a result the performance of a deeper network gets saturated or even starts deteriorating rapidly. ResNet’s popularity comes with its ability to overcome this issue in several ways. The center thought of ResNet is presenting a supposed "identity shortcut connection" that avoids at least one layers, as shown in the figure 2.5. [12]
The authors of [12] assert that stacking of layers ought not debase the system’s exhibition since we could just stack identity mappings i.e., layers that basically don’t perform anything, and the outcome would be the equivalent. The hypothesis is that letting the stacked layers fit a residual mapping is easier than letting these layers directly fit the desired underlying mapping. The residual block explicitly allows it to do so. A building block is defined as:

$$y = F(x, W_i) + x \tag{2.1}$$

When the input and output are of the same dimension, the identity shortcuts (2.1) can be directly used. When the dimensions increase, we consider two options: (A) The shortcut still performs identity mapping, for increased dimensions we pad extra zeroes (B) The projection shortcut is used to match dimensions (done by 1×1 convolutions).

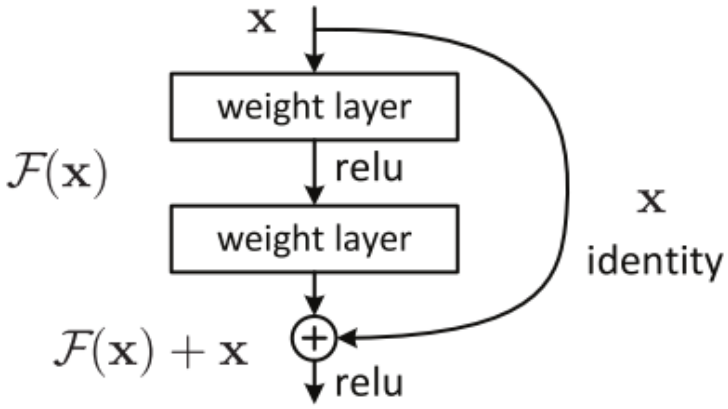


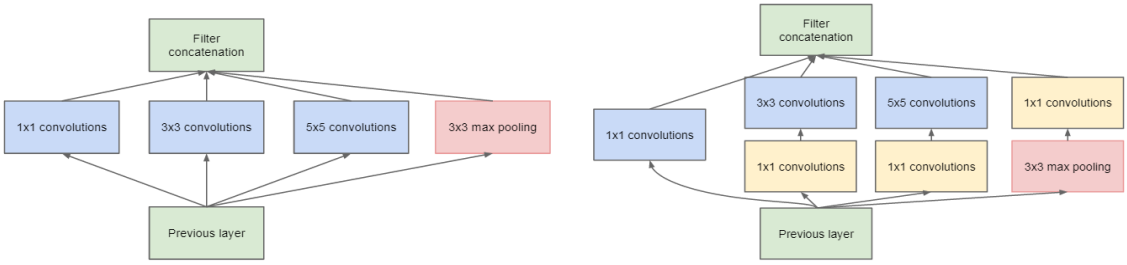
Figure 2.5: A residual block

Inception

Before the existence of Inception, the performance of Neural Networks was improved by making the networks deeper. These have a high chance of overfitting and can become computationally expensive. A way of solving these problems is to shift from fully connected to sparsely connect architectures.

[13]The Inception network's design depends on discovering how an optimal local sparse structure in a convolutional vision system can be approximated and covered by promptly accessible dense parts. Instead of deciding on a filter size, inception stacks filters of different sizes on top of each other. The outputs of these layers are then concatenated to be the input of the next layer Convolutions of size 1×1, 3×3 and 5×5 are used to avoid misalignment of clusters spatially. A larger size of filters will cause less number of clusters which spatially spread over the region. Inception essentially lets the network itself decide the best filter size. As a result, Inception works well with both, images where the subject is located globally and in which they are localised to a region in the image.

Since the network can become computationally expensive, 1×1 convolutions are used before applying the others for dimension reduction. A single Inception module is illustrated in 2.6. A complete network has 22 such layers without pooling layers. Overall the network has about 100 layers.



(a) Inception module, naïve version (b) Inception module with dimension reductions

Figure 2.6: Inception Modules

To ensure the effective propagation of the gradient back through all the layers, additional auxiliary classifiers are connected to the network. This addition is based on the insight that the middle layers of the network should be discriminating. During training, the loss from these classifiers is added to the overall loss with decreased weight. During inference, these classifiers are not used.

CHAPTER 3

MATERIALS AND METHODS

3.1 SYSTEM DESIGN

3.1.1 BLOCK DIAGRAM

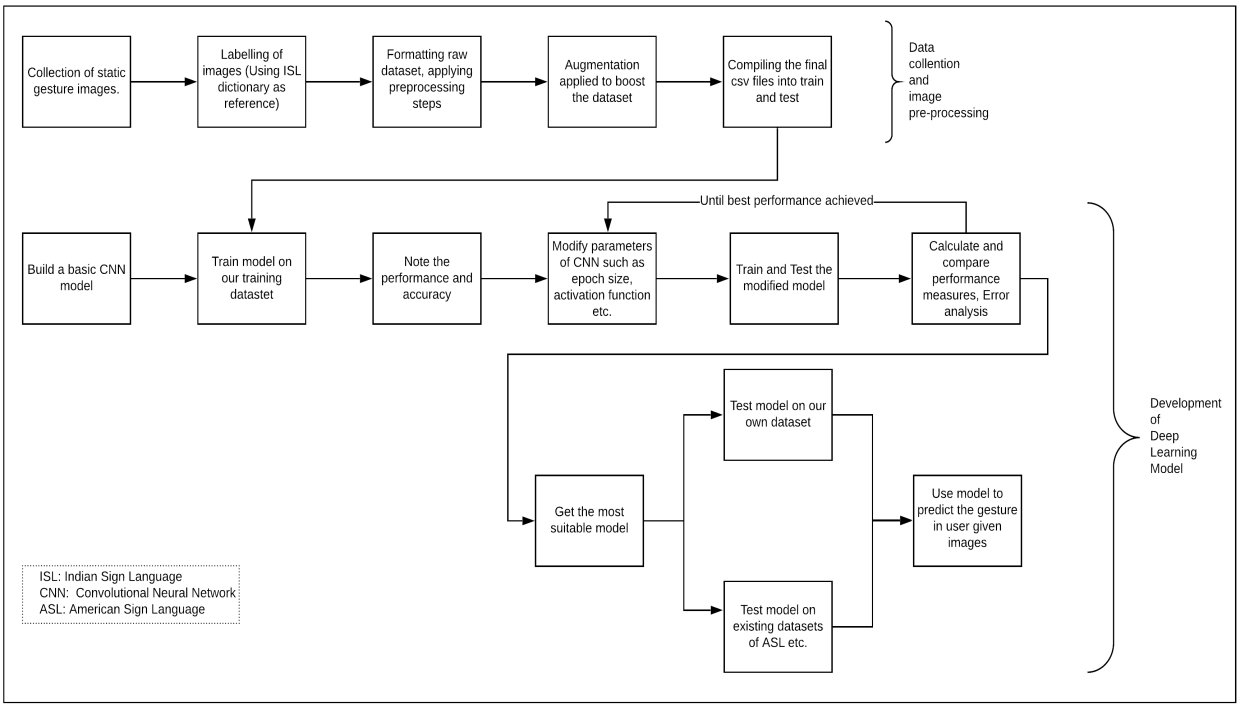


Figure 3.1: Block Diagram

3.1.2 DATA FLOW DIAGRAM

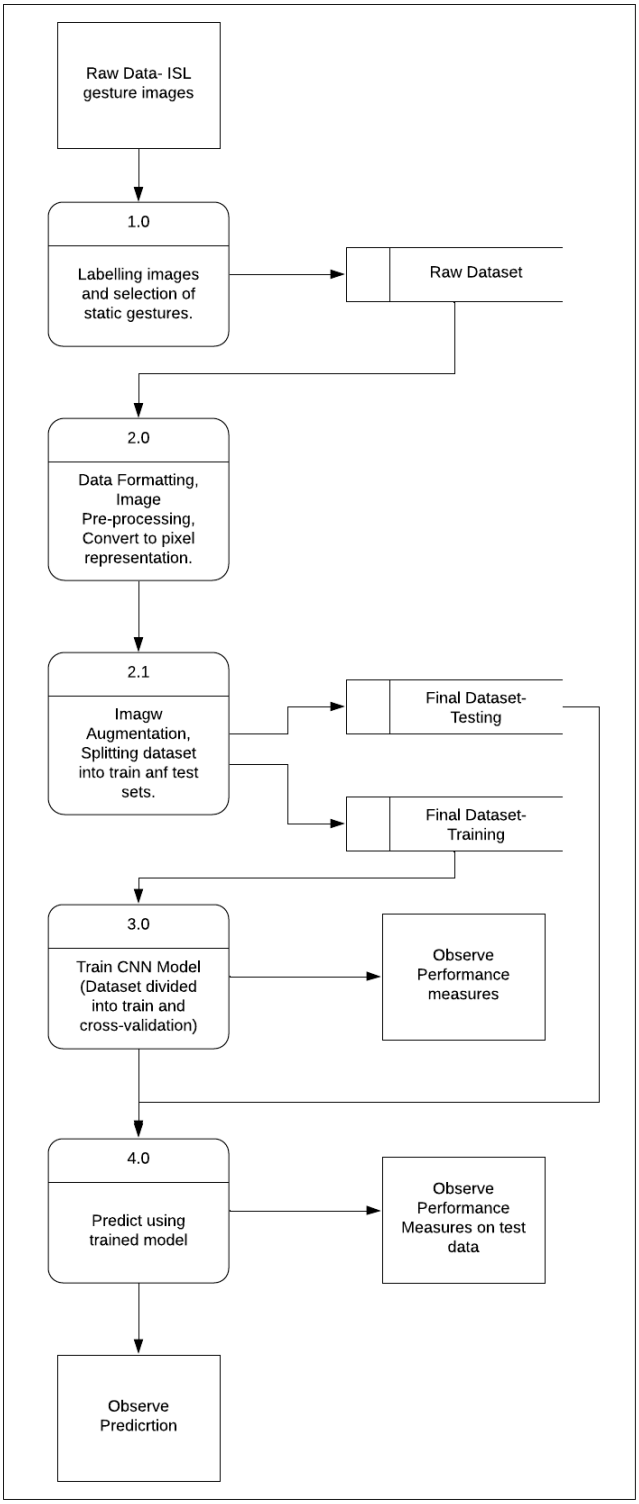


Figure 3.2: Data Flow Diagram

3.2 IMPLEMENTATION

3.2.1 DATASET GENERATION AND IMAGE PRE-PROCESSING

Due to the dearth of a comprehensive dataset for ISL, we needed to start from scratch. Once the images had been gathered and labelled, they needed to undergo Data Preparation to format them into a cohesive dataset ready to be trained by the models. Dataset preparation is a very important and time-consuming step which involves everything from the deciding what part of the data is relevant to building a training set. Our goal was to do two things:

1. Create a cohesive dataset to be used by anyone working on ISL
2. Make sure this dataset can be expanded as required

Keeping this in mind, the approach was to decide which IP techniques were required and which could be skipped. Initial approach was to use the standard process of greyscaling, threshholding segmentation. However, Deep Learning frameworks can be modelled on raw data. So the decided approach was to pre-train the models on a large existing dataset and then use it on the smaller set of data generated by us. The datasets for ASL were taken up as a reference point to create the ISL dataset. [14]

A. Image Generation

Our dataset has 278 images from a standard dataset curated by us, along with 90 images with variations in lighting, color, background, human signage differences and generated over DSLR, Webcam and mobile cameras. Our aim was to incorporate images in the dataset which would be very similar to the ones which will be fed by users, to increase authenticity and accuracy in recognition.(refer to Figure 3.3) At the inception of this project, there were no datasets available for ISL, so using the ISLRTC dictionary as a reference for labeling, we worked on classifying signs which were similar in ASL and ISL while working on generating our own (see: Code Phases for more details).

B. Data Formatting and Cleaning

To standardise the images, we used the Portable Network Graphics format as it is the best when dealing with true-color images that cannot work well with JPEG compres-

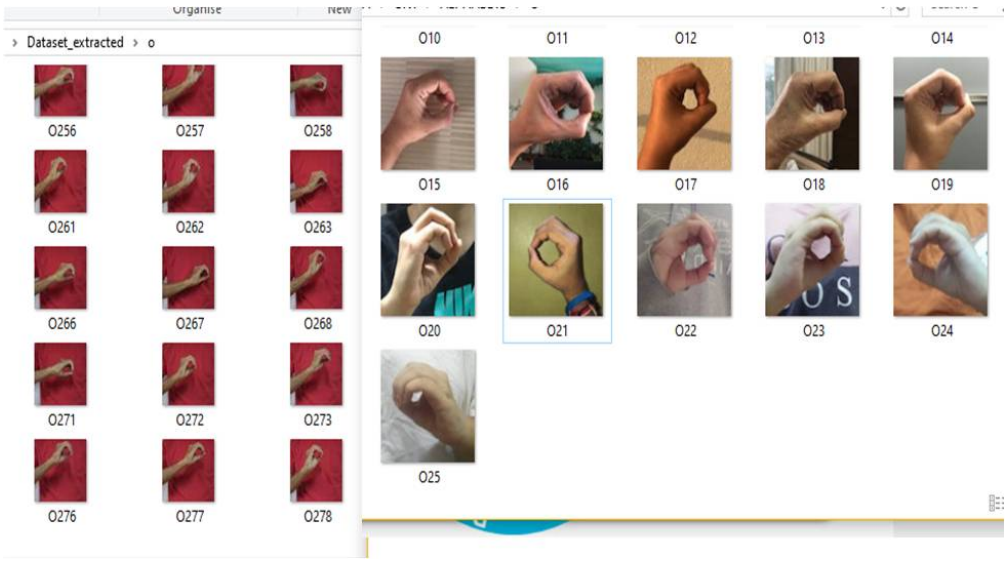


Figure 3.3: Left: Alphabet O from standard dataset
Right: Alphabet O generated by us (notice the variations, also includes left-handed signage)

sion, which is perfect for this project which emphasises on color input and requires quality images. PNG uses a lossless image compression technique, hence no information is lost and the image is retrieved keeping the same quality. A disadvantage due to this is that the files are larger in size. So, to accommodate a large number of images, we had to remove redundant data from the images. Here, we only require the hand gestures, so we used cropping to remove noise (in this case, the background). Cropping an image also reduces the computational time while training the classifier. The initial crop size was 50x50 pixels, but it proved to be very grainy to be deciphered by the model during training. The images were then resized to a size of 100x100 pixels for increased clarity upon reconstruction.

C. Augmentation

As the amount of images were less, the model was overfitting. There were also biases caused by classes which had more images so the solution was to augment the images and create a dataset with uniform number of images per class. The augmentation techniques selected were rotation and greyscaling, and saturation. Greyscaling was discarded due to the fact that the model reconstructed all the csv images into greyscale, hence the image wouldn't be a different image. Upon augmentation, we now have a cohesive dataset of 600 images per class (alphabet), making it the first of one of the most reliable

and diverse ISL datasets containing about 15000 images.

D. Dataset Creation

Once the dataset was labelled and formatted and ready to be complied, the images were split into a train:test ratio of 80:20, converted into csv values using image representation and shuffled to generate 2 csv files of train and test alphabets ready to be used by the model. These files contains 10,000 (100x100) columns containing RGB value (0,255) of every pixel, and an extra column containing the label. Each row stores data of 1 image. Some of the Python libraries and modules used include pandas, os and PIL. The whole Pre-Processing and Dataset Generation process is shown in Figure 3.4).

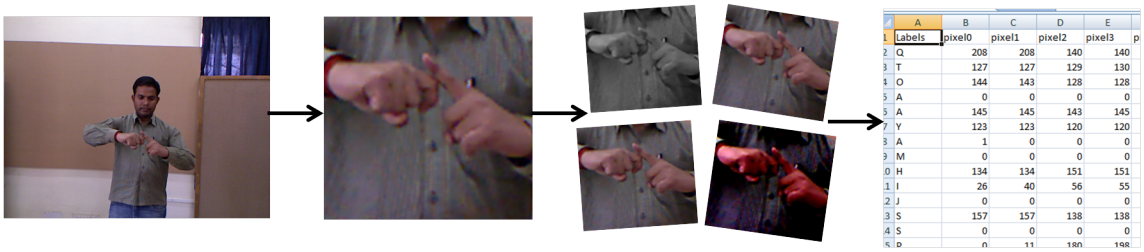


Figure 3.4: **Image 1:** Raw images are labeled and grouped using ISLRTC dictionary.
Image 2: Image undergoes formatting and standardisation. (PNG conversion, cropping, resizing)
Image 3: Augmentation techniques like rotation, greyscaling and saturation are used to increase the dataset.
Image 4: Dataset is split and shuffled, and converted to csv files of train and test.

AUTOMATED DATASET GENERATION SYSTEM

Along with creating a recognition system, we also want to support further research on ISL, along with the ability to add to the dataset. So we created an automated pre-processing system which can do the following Pre-Processing steps:

1. labeling
2. resizing
3. augmenting
4. splitting
5. shuffling

It finally creates their csv files in a single code with no intervention, provided that the images are cropped and grouped according to classes. This system, while generated keeping in mind the project requirements, can be used to generate image datasets and can be customised for any machine learning and deep learning project.

3.2.2 MODEL CREATION

We began by exploring the different layers of CNN to find the best working architecture for our problem statement. Starting with a 2 layered sequential model having one CNN and one FCN(called as Dense layer). This basic model performed poorly, giving both low train accuracy and low test accuracy. We iteratively applied different variations of layers, number of layers, optimizing function, number of epochs, loss functions, activation function, pooling layers and batch sizes. The model creation step consists of various phases, each phase consisting of an architecture of the model used for training the dataset. Every subsequent phase consists of an architecture built by improvising on the shortcomings of the previous architecture.

Code Phase 1.0

After experimenting with various parameters, we built an intermediate model to classify a few static signs of ISL. The number of images in the training dataset were 1499 and the number of images in the test dataset were 399. These included the C, L, O, V and W. They were obtained from an internet available ASL dataset as these images intersected in both, ISL and ASL. Increasing the size of the dataset was leaving no room for any improvement. Even though the accuracy is near perfect in the case of a very large dataset, there was no scope for the model to make mistakes and learn. Hence, when faced with a new image, the model was giving incorrect prediction. Therefore, we learned that when the CNN is small and the number of classes is also small, it is advisable to not have an enormous dataset. The images in the dataset were resized to (28,28) and corresponding to this size, we took the number of layers as 3. The ReLU activation function is used in the hidden layers as it is computationally cheaper than sigmoid, which has to perform complex exponential operations. It also aids with the vanishing gradient problem. Overall it is proven to show better convergence of the network. Softmax activation is used for output layer as it is best suited for multiclass

classifications.

Brief experiments and their observations are explained below:

- In the case of average pooling, there was considerate fluctuations in the accuracy and loss values. Fluctuation in accuracy implies that there is a certain amount of randomness in predicting the labels [15]. Max pooling does not give this fluctuation and hence, we opted to use max pooling for further experiments.
- ReLu activation function was used in the layers of CNN as Sigmoid function gives a higher loss value.
- The final epoch value was maintained at 15 as decreasing or increasing the number of epochs resulted in overfitting of the data.
- We experimented with two loss functions, Kullback Leibler Divergence and Categorical Crossentropy. We observed that in the case of Kullback Leibler Divergence, there is an unpredictable rise and fall of accuracy and loss values. Hence, we decided to go for Categorical Crossentropy.

The model architecture thus obtained is as follows:-

Sr. No.	Layer	Filter Size	Activation Func.	No. of Filters/Units	Output Size
1	Conv2D	(3,3)	relu	64	(None, 26, 26, 64)
2	MaxPool	(2,2)	NA	64	(None, 13, 13, 64)
3	Conv2D	(3,3)	relu	64	(None, 11, 11, 64)
4	MaxPool	(2,2)	NA	64	(None, 5, 5, 64)
5	Conv2D	(3,3)	relu	64	(None, 3, 3, 64)
6	Flatten	NA	NA	NA	(None, 576)
7	Dense	NA	relu	64	(None, 64)
8	Dense	NA	softmax	5	(None, 5)

Table 3.1: Intermediate Architecture 1.0

The following results were obtained from this model.

1. Training accuracy: 1.0000
2. Training loss: 9.3412e-04
3. Validation accuracy: 1.000
4. Validation loss: 0.0021
5. Testing accuracy: 0.9524
6. Testing loss: 0.0996

Code Phase 2.0

In this phase, we have used the basic architecture obtained in 3.2.2 and modified different aspects such as dataset size, number of layers etc. We started this phase by analysing the problems encountered in 3.2.2 such as training time complexity high, incorrect prediction, limited scope for the model to learn etc. We began expanding the dataset and working on exploring different augmentation techniques. Also, we reconstructed the dataset images for better image representation. The dataset used in this phase had 2799 training images and 999 testing images, consisting the letters C, K, L, O, V and W. It contained letters from the previous dataset along with images of our own. We experimented with increasing the number of layers in the model, ergo the model complexity. However, the number of layers could not be increased by more than one as the image size would not allow it. Furthermore, increasing the complexity of the model with a small dataset would result in overfitting of the data. Hence, the final architecture obtained was as follows:

Sr. No.	Layer	Filter Size	Activation Func.	No. of Filters/Units	Output Size
1	Conv2D	(3,3)	relu	64	(None, 98, 98, 64)
2	MaxPool	(2,2)	NA	64	(None, 49, 49, 64)
3	Conv2D	(3,3)	relu	64	(None, 47, 47, 64)
4	MaxPool	(2,2)	NA	64	(None, 23, 23, 64)
5	Conv2D	(3,3)	relu	64	(None, 21, 21, 64)
6	MaxPool	(2,2)	NA	64	(None, 10, 10, 64)
7	Conv2D	(3,3)	relu	64	(None, 8, 8, 64)
8	Flatten	NA	NA	NA	(None, 4096)
9	Dense	NA	relu	64	(None, 64)
10	Dense	NA	softmax	6	(None, 6)

Table 3.2: Intermediate Architecture 2.0

The following results were obtained from this model.

1. Training accuracy: 0.9489
2. Training loss: 0.1435
3. Validation accuracy: 0.9392
4. Validation loss: 0.1416
5. Testing accuracy: 0.9079
6. Testing loss: 0.1573

Code Phase 3.0

For this phase, we acquired a dataset from another researcher, containing all the alphabets of ISL. It consisted around 6000 training images and 1400 testing images of size $50 * 50$ each. The model obtained in 3.2.2 had to be adjusted to train the new image size and new number of output neurons(26), and was tested on the new dataset. While the model achieved a good accuracy (mentioned ahead), it did not give good predictions on new images.

The following results were obtained from this model.

1. Training accuracy: 0.9836
2. Training loss: 0.0835
3. Validation accuracy: 0.9690
4. Validation loss: 0.1574
5. Testing accuracy: 0.9672
6. Testing loss: 0.2322

We inspected the following to determine the cause:-

- **Image size**: Attributing the same to the smaller image size(i.e. decreased number features to be learnt/image too blur), we interpolated the images to size $100 * 100$ and changed the model to Phase 2 (3.2.2) again. This approach only improved the accuracy of the model, not the predictions.
- **Changing size of the model/architecture of the model**: Changing the layers or the number of layers. The model accuracy improved, but did not aid the wrong predictions.
- **Bias or Variance issues**: Inability to predict new images is an indication of overfitting or high variance. However, the values of training, validation and test accuracy and loss did not indicate any such problem. We still tried adding dropout, regularisation techniques. However, they did not have any effect on the model.

On further inspection, we determined that the problem was that dataset lacked variation in terms background, noise and lighting. It only contained variation in orientation. Since all three, the test, validation and test sets come from the same distribution of images, the model performed well on them but not on new images.

Code Phase 4.0

For the final coding phase, we improved the quality of our dataset by adding images with different background, noise and lighting. We manually clicked and collected these images from our family and friends and augmented them so that we have substantial noise in the dataset. The new and improved dataset consisted of 10114 training images and 2522 testing images. The model gave good accuracy and fairly good predictions with this dataset but it had to be fine tuned further to give better results. We also carried out error analysis to determine which of the alphabets were wrongly predicted by the model. We conducted the following experiments on the model to achieve the best possible results in terms of accuracy, loss and predictions:

- **Layers (Convolution and Pooling)**: To experiment with different number of convolution and maxpool layers, we increased/decreased both these layers and noted the change in the number of wrongly predicted images. After increasing one convolution layer and decreasing one maxpool layer, the number of wrong predictions increased. Similar behaviour was observed when two convolutions layers were increased and one maxpool layer was decreased. This takes place due to overfitting. Increasing convolution layers corresponds to increase in model complexity. We add maxpool layers to decrease the number of parameters, thus preventing overfitting of the model. By decreasing maxpool layers, we are essentially promoting overfitting, which is not desirable. Hence, we kept the number of maxpool layers constant and increased two convolution layers. This significantly decreased the number of wrongly predicted images.

- **Maxpool Strides**: Changing the stride of the maxpool layer was inspired by the maxpool layers in VGG model. The model trained with the stride = 2 proved to have smoother training.

- **Early Stopping and Epochs**: To prevent the overfitting of the model, a stopping function was created. The function was a callback function, which was called inside the training function. It monitors the training accuracy of the model and stops the training when it becomes 90%. While this method prevented overfitting, the model was reaching sufficient validation and test accuracy. To fix that, we experimented with increasing the cutoff accuracy so that we achieve a balance between not overfitting and enough training time. The cutoff was set at 99%.

- **Regularisation and Bias Regularisation**: We used regularisation to reduce overfitting and help the model generalise better. In our model, we have used L2 regularisation, also known as "weight decay". A very high value of coefficient was decreasing the model accuracy considerably. Hence, we set a lower coefficient value so that the accuracy is not hampered while reducing overfitting. Moreover, we added regularisation to only two of the convolution layers as adding regularisation to all the layers would decrease the accuracy of the model.

- **Batch Size**: We used Trial and Error method to fix the batch size at 512. Lower batch sizes had a lower accuracy and higher sizes caused exhaustion of memory resource.
- **Dropout**: It is a regularisation technique that randomly ignores some nodes of the previous layer. However dropout does not perform well when placed between convolutional layers since they share filters and have a local-connectivity architecture. This leads to the reduction in the number of parameters and already reduces the possibility to overfit[16]. Placing it between FCN layers caused the training time to increase significantly and the model to heavily underfit. Hence the dropout layer was discarded.
- **Dataset Split**: The model performed increasingly better the time more data was provided for training. Hence, we split the dataset into 80-20 train to test ratio. The train was further split into 90-10 ratio to create train and validation sets. Making the whole split of the ratio 70-10-20. If we trained the model on less than 70% of the total data, the predictions of the model were hampered.

Error Analysis: Along with the tracking of the accuracies and the losses, we also kept track of the wrongly predicted alphabet images of the test set. We tracked the number of wrong predictions as well as the specific alphabets that were predicted wrong. When making any changes to the model, we also checked the effects on the errors. A few letters had a higher number of prediction errors, but they were resolved with more amount of training data.

After evaluating these parameters, we finalised on the following model architecture:

Sr. No.	Layer	Filter Size	Activation Func.	No. of Filters/Units	Output Size
1	Conv2D	(3,3)	relu	64	(None, 98, 98, 64)
2	Conv2D	(3,3)	relu	64	(None, 96, 96, 64)
3	MaxPool	(2,2)	NA	64	(None, 48, 48, 64)
4	Conv2D	(3,3)	relu	64	(None, 46, 46, 64)
5	Conv2D	(3,3)	relu	64	(None, 44, 44, 64)
6	MaxPool	(2,2)	NA	64	(None, 22, 22, 64)
7	Conv2D	(3,3)	relu	64	(None, 20, 20, 64)
9	Conv2D	(3,3)	relu	64	(None, 18, 18, 64)
10	MaxPool	(2,2)	NA	64	(None, 9, 9, 64)
11	Flatten	NA	NA	NA	(None, 5184)
12	Dense	NA	softmax	26	(None, 26)

Table 3.3: Intermediate Architecture 4.0

Using this architecture we achieved the following results:

1. Training accuracy: 0.9985
2. Training loss: 0.0135

3. Validation accuracy: 0.9733
4. Validation loss: 0.1089
5. Testing accuracy: 0.9782
6. Testing loss: 0.0836

NOTE: The reason the model has higher validation loss than test loss is due to the way the training function splits the dataset into train and validation sets. The model has similar validation and test loss when the training and validation sets are equally distributed across all classes. The training function splits the dataset into the respective sets randomly, causing the discrepancy.

3.3 SOFTWARE TOOLS, TECHNOLOGIES USED

While not an exhaustive list, here are few of the libraries and frameworks we have used in our project.

- Python: Python is a widely used high-level programming language. It is an object-oriented, interpreted and interactive programming language. It consists of modules, classes, very high-level of dynamic data types and has a very clear syntax. Python has great data-handling capacity. It is the preferred choice for machine learning applications because of the availability of various libraries such as tensorflow, keras, pandas, numpy etc.
- PIL: PIL Python library is known in the current versions as Pillow. It is and is free and open-source. Python Imagine Library (PIL) offers image processing capabilities as well as graphics. Is adds support for the opening, manipulating, and saving multiple different image file formats.
- TensorFlow: It is a library released by Google for fast numerical computing. It offers a foundation which can be used to build various deep learning models. TensorFlow can run on several kinds of systems like single CPU systems, GPUs, mobile devices and a large scale distributed system as well.
- Keras: It is a high-level NN API written in Python. It is capable of running on top of TensorFlow, CNTK, or Theano. Fast experimentation was the focus during its development. It allows for fast and easy prototyping through user friendliness, modularity, and extensibility. Keras supports CNN, Recurrent Neural Network (RNN), as well as combinations of the two. Moreover, it runs seamlessly on both, CPU and GPU.
- Pandas: It is an opensource library that helps users to perform data manipulation in python. It offers operations for manipulating numerical series and time series. It offers data structures as well Pandas is built on top of NumPy which means that it needs the presence of NumPy to work.

- NumPy: NumPy stands for numerical python and consists of multidimensional arrays and resources and routines to manipulate them. They help in performing mathematical and logical operations on data.
- OpenCV: OpenCV-Python is a Python wrapper for the original OpenCV C++ implementation. OpenCV-Python uses Numpy(library for numerical operations). All the OpenCV arrays are converted to Numpy arrays and vice versa.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Model Results

We plotted the graphs of training and validation accuracy and loss. The graph of training and validation accuracy against the number of epochs is shown below.

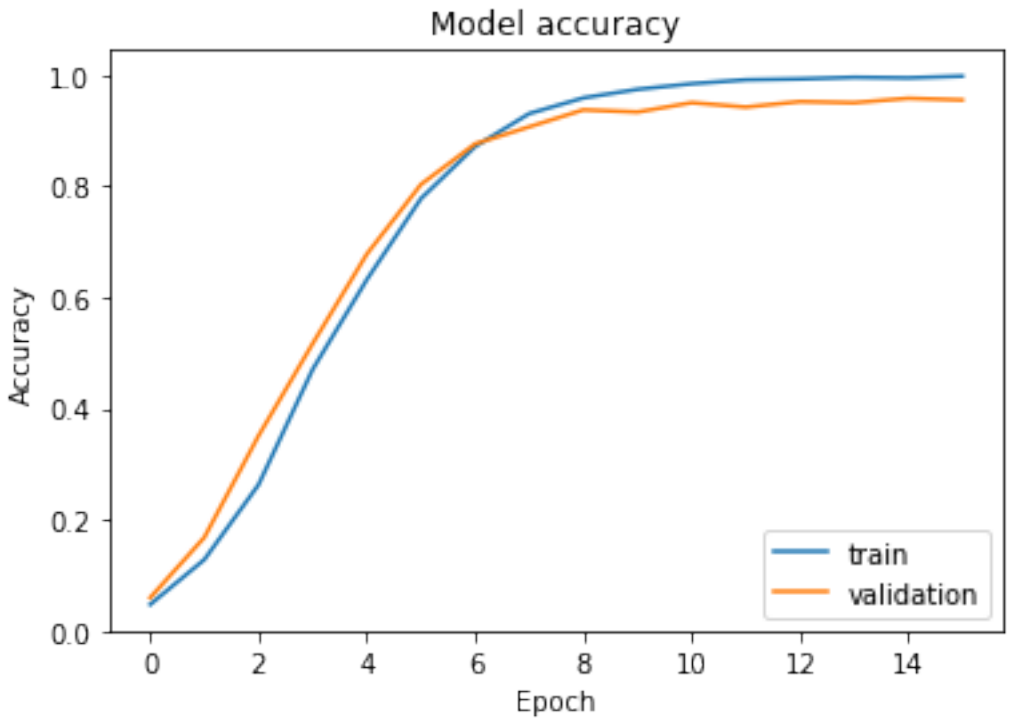


Figure 4.1: Model Accuracy

The graph of training and validation loss against the number of epochs is shown below.

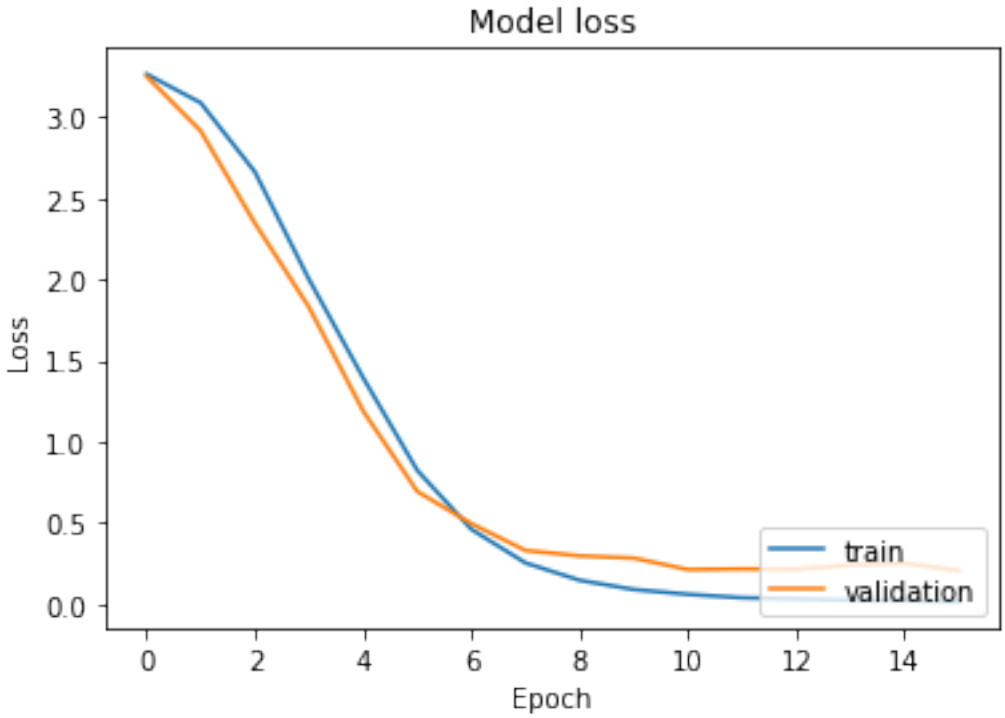


Figure 4.2: Model Loss

From the graphs we can conclude that the model does not overfit as there is little difference between the training and validation accuracy and loss values.

To analyse the performance of our multi-label classification model, we calculated a few metrics.

- Fscore: The fscore is a measure of the accuracy of the classification algorithm. It uses both, precision and recall. The fscore ranges between the values 0 (inadequate) and 1(finest). The fscore of our model is 0.9806
- Multi-label confusion matrix: Each of the nested matrices in the multi-label confusion matrix represents a class (alphabet in our case). Multiclass data will be treated as if binarized under a one-vs-rest transformation. The confusion matrix obtained for our model is as follows:

  array([[[2424, 1], [3, 94]],

  [[2425, 0], [0, 97]],

  [[2425, 0], [1, 96]],

  [[2425, 0], [1, 96]],

  [[2422, 3], [2, 95]],

  [[2424, 1], [1, 96]],

  [[2425, 0], [0, 97]],

  [[2419, 6], [1, 96]],

  [[2422, 3], [3, 94]],

[[2425, 0], [1, 96]],
[[2423, 2], [2, 95]],
[[2422, 3], [2, 95]],
[[2417, 8], [6, 91]],
[[2415, 10], [4, 93]],
[[2425, 0], [0, 97]],
[[2425, 0], [0, 97]],
[[2422, 3], [2, 95]],
[[2425, 0], [1, 96]],
[[2424, 1], [4, 93]],
[[2423, 2], [4, 93]],
[[2424, 1], [1, 96]],
[[2425, 0], [1, 96]],
[[2424, 1], [0, 97]],
[[2424, 1], [9, 88]],
[[2425, 0], [0, 97]],
[[2422, 3], [0, 97]]])

- Classification report: The complete classification report of the algorithm was generated and it gave the following values:

	precision	recall	f1-score	support
0	0.99	0.97	0.98	97
1	1.00	1.00	1.00	97
2	1.00	0.99	0.99	97
3	1.00	0.99	0.99	97
4	0.97	0.98	0.97	97
5	0.99	0.99	0.99	97
6	1.00	1.00	1.00	97
7	0.94	0.99	0.96	97
8	0.97	0.97	0.97	97
9	1.00	0.99	0.99	97
10	0.98	0.98	0.98	97
11	0.97	0.98	0.97	97
12	0.92	0.94	0.93	97
13	0.90	0.96	0.93	97
14	1.00	1.00	1.00	97
15	1.00	1.00	1.00	97
16	0.97	0.98	0.97	97
17	1.00	0.99	0.99	97
18	0.99	0.96	0.97	97
19	0.98	0.96	0.97	97
20	0.99	0.99	0.99	97
21	1.00	0.99	0.99	97
22	0.99	1.00	0.99	97
23	0.99	0.91	0.95	97
24	1.00	1.00	1.00	97
25	0.97	1.00	0.98	97
accuracy			0.98	2522
macro avg	0.98	0.98	0.98	2522
weighted avg	0.98	0.98	0.98	2522

Figure 4.3: Classification Report

- Receiver Operating Characteristic (ROC): ROC is used to evaluate classifier output quality. ROC score is usually between 0.5 to 1 where, 0.5 means bad classifier and 1 means good classifier. For our model, the value is 0.9898

4.2 Performance of Existing Models and Comparison

4.2.1 VGG

The VGG19 model from the Keras API was used to train on the dataset. VGG19 model consists of 26 layer. Layers include Convolution layers and maxpool layers. It has 2 fully connected layers and the output layer.

The model was trained using transfer learning. The 'ImageNet' pre-trained weights were used. The layers of the base model were frozen(stopped from tranining), except the last 10 layers. Additionally, we added a dropout layer and an output layer to the model to make it suitable to classify 26 alphabets.
After training the model achieved the following results-

1. Training accuracy: 0.9927
2. Training loss: 0.0307
3. Validation accuracy: 0.9901
4. Validation loss: 0.0411
5. Testing accuracy: 0.9920
6. Testing loss: 0.0306

The model achieved a high accuracy in all three, train, validation and test sets. But, the model overfitted on the data and the predictions on new images were not correct. The issue can be solved by stopping the training earlier, but the accuracy does not reach 99% and have to stop much earlier to stop overfitting. Another issue with the model is the training time which was around 12 minutes. Overall, the model was too deep to learn the model without overfitting.

4.2.2 ResNet

A ResNet model was built in Keras using the pretrained ResNet-50 library. The "weights" parameter was set to None so as to use random weights to initialise the model. A global average pooling layer and a dense layer was added along with dropout regularisation to the pretrained model. A softmax activation function is used along with Adam optimization with a learning rate of 0.0001. The learning rate was decided after conducting

various experiments with different values.
The model was trained for 35 epochs on a batch size of 512 which also, was determined after experimenting with various values.
After training, the model achieved the following:

1. Training accuracy: 0.8532
2. Training loss: 0.1502
3. Validation accuracy: 0.8367
4. Validation loss: 0.2091
5. Testing accuracy: 0.8105
6. Testing loss: 0.2740

The model achieved a reasonable accuracy in all three training, validation and test sets. Low accuracy can be attributed to the random initialization of the weights. The predictions on new images were not accurate, this can be due to overfitting of the model. Upon researching, we found out that ResNet does not give a high accuracy if the dataset is not huge enough(along the lines of Imagenet) and images are not of very high resolution. Overall, the model was too specific to deal with our application.

4.2.3 Inception

Similar to VGG, the InceptionV3 model of Keras API was used. It has 159 layers consisting of various layers. Pre-trained 'ImageNet' weights were used for transfer learning on this model too. We froze the first 20 layers of the model and trained the rest. We also added 1 FCN later, 1 dropout layer and the output later will output of 26 classes.

The images had to be interpolated to size $200 * 200$ because the model performed better on images of larger size.

Following are the results of the model-

1. Training accuracy: 0.9936
2. Training loss: 0.0454
3. Validation accuracy: 0.7667
4. Validation loss: 0.7948

5. Testing accuracy: 0.7680
6. Testing loss: 0.7835

Being a very complex model, the number of samples in the datasets were to less for it. As we can see from the test and validation losses in the results, the model has very high variance. The training also requires high amount of RAM and processing power. The time taken for training was also very long, around 45 minutes.

4.2.4 Comparison

Model Name	Train	Validation	Test
CNN	99.85	97.33	97.82
VGG19	99.27	99.01	99.20
ResNet	85.32	83.67	81.05
InceptionV3	99.36	76.67	76.80

Table 4.1: Accuracy Comparison(in %)

Model Name	Train	Validation	Test
CNN	1.35	10.98	8.36
VGG19	3.07	4.11	3.06
ResNet	15.02	20.91	27.40
InceptionV3	4.54	79.48	78.35

Table 4.2: Loss Comparison(in %)

Model Name	Training Time
CNN	2.44
VGG19	11.85
ResNet	37.52
InceptionV3	43.92

Table 4.3: Training Time Comparison(in mins)

CHAPTER 5

CONCLUSION

In this project, the recognition and translation system for the newly defined ISL has been designed. Due to the unavailability of a standard and well-labeled data set, little research work has been published on a deep learning model that works on ISL. This report discusses an automatic dataset generation system which can be used to generate image datasets for any machine learning or deep learning projects. Through this, we have created a dataset from scratch and are planning on releasing the ISL dataset to the research community to create awareness and promote further work on the eradication of communication barriers between the verbal and hearing impaired section of the society. In this project, we have built a CNN model from scratch that provides the best performance for our dataset. Various experiments were carried out to determine the best fit for our dataset. Meanwhile, we also studied models like Inception, ResNet and VGG and compared the performance of these models to our own. Our aim was to create a system that could recognise the static gestures in the alphabets and translate them into English for easy understanding. By completing this project we have successfully began the journey of decreasing the communication gap between the community of the hearing-impaired and the rest of the society.

REFERENCES

[1] P Subha Rajam and G Balakrishnan. Real time indian sign language recognition system to aid deaf-dumb people. In *2011 IEEE 13th International Conference on Communication Technology*, pages 737–742. IEEE, 2011.

[2] J Rekha, J Bhattacharya, and S Majumder. Shape, texture and local movement hand gesture features for indian sign language recognition. In *3rd International Conference on Trendz in Information Sciences & Computing (TISC2011)*, pages 30–35. IEEE, 2011.

[3] Vivek Bheda and Dianna Radpour. Using deep convolutional networks for gesture recognition in american sign language. *arXiv preprint arXiv:1710.06836*, 2017.

[4] Sirshendu Hore, Sankhadeep Chatterjee, V Santhi, Nilanjan Dey, Amira S Ashour, Valentina Emilia Balas, and Fuqian Shi. Indian sign language recognition using optimized neural networks. In *Information Technology and Intelligent Transportation Systems*, pages 553–563. Springer, 2017.

[5] Brandon Garcia and Sigberto Alarcon Viesca. Real-time american sign language recognition with convolutional neural networks. *Convolutional Neural Networks for Visual Recognition*, 2:225–232, 2016.

[6] altexsoft. Machine learning project structure: Stages, roles, and tools, 2018.

[7] Kajal B Borole, Bhagyashree D Patil, and HD Gadade. Gesture recognition using image processing and conversion to text and speech. *International Journal of Exploring Emerging Trends in Engineering (IJEETE)*, 3(02), 2016.

[8] Dr. Jeegar Trivedi Hemina Bhavsar. *Review on Feature Extraction methods of Image based Sign Language Recognition system*. PhD thesis, S.S.Agrawal Institute of Computer Science, Indian Journal of Computer Science and Engineering (IJCSE), 6 2017.

[9] Vivek Bheda and Dianna Radpour. Using deep convolutional networks for gesture recognition in american sign language. *CoRR*, abs/1710.06836, 2017.

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[11] Sasha Targ, Diogo Almeida, and Kevin Lyman. Resnet in resnet: Generalizing residual architectures. *arXiv preprint arXiv:1603.08029*, 2016.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[14] Josué Ibarra Molinas. Deep learning.

[15] Kaushik Gobindram Pasi and Sowmiyaraksha R Naik. Effect of parameter variations on accuracy of convolutional neural network. In *2016 International Conference on Computing, Analytics and Security Trends (CAST)*, pages 398–403. IEEE, 2016.

[16] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

ISL Project Report

ORIGINALITY REPORT

8%	6%	5%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.slideshare.net Internet Source	2%
2	refitsl.com Internet Source	1%
3	link.springer.com Internet Source	<1%
4	machinelearningmastery.com Internet Source	<1%
5	"Deep Learning in Healthcare", Springer Science and Business Media LLC, 2020 Publication	<1%
6	repository.tudelft.nl Internet Source	<1%
7	Kinnary Joshi, Kalind Karia, Jaykumar Patel, Sanjana Desai. "Library Stock Verification System using Artificial Neural Networks", 2018 International Conference on Smart City and Emerging Technology (ICSCET), 2018 Publication	<1%

8	www.scribd.com Internet Source	<1%
9	J. Rekha. "Shape, texture and local movement hand gesture features for Indian Sign Language recognition", 3rd International Conference on Trendz in Information Sciences & Computing (TISC2011), 12/2011 Publication	<1%
10	Fang Fang, Zhen Wu, Luchen Zhang, Shi Wang, Cungen Cao. "A fusion method of text categorization based on key sentence extraction and neural network", 2017 2nd International Conference on Knowledge Engineering and Applications (ICKEA), 2017 Publication	<1%
11	Vaishnavi Mande, Mandar Lakhe. "Automatic Video Processing Based on IoT Using Raspberry Pi", 2018 3rd International Conference for Convergence in Technology (I2CT), 2018 Publication	<1%
12	Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. "Deep Residual Learning for Image Recognition", 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016 Publication	<1%

13	Venkata Chirra, Srinivasulu ReddyUyyala, Venkata KishoreKolli. "Deep CNN: A Machine Learning Approach for Driver Drowsiness Detection Based on Eye State", Revue d'Intelligence Artificielle, 2019 Publication	<1%
14	Ritika Gulyani. "Chapter 16 Examining the Question of Deaf Education within Disability Studies in Delhi, India", Springer Science and Business Media LLC, 2020 Publication	<1%
15	arxiv.org Internet Source	<1%
16	ermt.net Internet Source	<1%
17	livrepository.liverpool.ac.uk Internet Source	<1%
18	Erik Cambria, Amir Hussain. "Sentic Computing", Springer Science and Business Media LLC, 2012 Publication	<1%
19	eprints.utar.edu.my Internet Source	<1%
20	trepo.tuni.fi Internet Source	<1%

21	www.scss.tcd.ie Internet Source	<1%
22	www.ijcse.com Internet Source	<1%
23	Futane, Pravin R, and Rajiv V. Dharaskar. ""“Hasta Mudra”: An interpretation of Indian sign hand gestures", 2011 3rd International Conference on Electronics Computer Technology, 2011. Publication	<1%
24	raiith.iith.ac.in Internet Source	<1%
25	www.coep.org.in Internet Source	<1%
26	educationnewsupdates.blogspot.com Internet Source	<1%
27	brage.bibsys.no Internet Source	<1%
28	ibgnews.com Internet Source	<1%
29	"Proceedings of International Conference on Artificial Intelligence and Applications", Springer Science and Business Media LLC, 2021 Publication	<1%

30	docplayer.net Internet Source	<1%
31	www.aclweb.org Internet Source	<1%
32	"Artificial Neural Networks and Machine Learning – ICANN 2019: Theoretical Neural Computation", Springer Science and Business Media LLC, 2019 Publication	<1%
33	cs231n.stanford.edu Internet Source	<1%
34	eprints.nottingham.ac.uk Internet Source	<1%
35	"Interpretation of Formal Semantics from Hand Gesture to Text using Proficient Contour Tracing Technique", International Journal of Recent Technology and Engineering, 2019 Publication	<1%
36	"Advances in Computing Applications", Springer Science and Business Media LLC, 2016 Publication	<1%
37	Z. Wu, X. Chen, Y. Gao, Y. Li. "RAPID TARGET DETECTION IN HIGH RESOLUTION REMOTE SENSING IMAGES USING YOLO MODEL", ISPRS - International Archives of the	<1%

	Photogrammetry, Remote Sensing and Spatial Information Sciences, 2018 Publication	
38	www.mdpi.com Internet Source	<1%
39	Hovannes Kulhandjian, Prakshi Sharma, Michel Kulhandjian, Claude D'Amours. "Sign Language Gesture Recognition Using Doppler Radar and Deep Learning", 2019 IEEE Globecom Workshops (GC Wkshps), 2019 Publication	<1%
40	mi.eng.cam.ac.uk Internet Source	<1%
41	Gong, Chen, Zhang, Zhang, Wang, Guan, Wang. "A Novel Deep Learning Method for Intelligent Fault Diagnosis of Rotating Machinery Based on Improved CNN-SVM and Multichannel Data Fusion", Sensors, 2019 Publication	<1%
42	export.arxiv.org Internet Source	<1%
43	www.ripublication.com Internet Source	<1%
44	P. Subha Rajam, G. Balakrishnan. "Real time Indian Sign Language Recognition System to aid deaf-dumb people", 2011 IEEE 13th	<1%

International Conference on Communication Technology, 2011
Publication

45	mafiadoc.com Internet Source	<1%
46	P.K. Athira, C.J. Sruthi, A. Lijiya. "A Signer Independent Sign Language Recognition with Co-articulation Elimination from Live Videos: An Indian Scenario", Journal of King Saud University - Computer and Information Sciences, 2019 Publication	<1%
47	Jingpeng Zhai, Weiran Shen, Ishwar Singh, Tom Wanyama, Zhen Gao. "A Review of the Evolution of Deep Learning Architectures and Comparison of their Performances for Histopathologic Cancer Detection", Procedia Manufacturing, 2020 Publication	<1%
48	Lecture Notes in Computer Science, 2015. Publication	<1%
49	Ali Serener, Sertan Serte. "Transfer Learning for Early and Advanced Glaucoma Detection with Convolutional Neural Networks", 2019 Medical Technologies Congress (TIPTEKNO), 2019 Publication	<1%

Amir Erfan Eshratifar, Amirhossein Esmaili,

50	Massoud Pedram. "Towards Collaborative Intelligence Friendly Architectures for Deep Learning", 20th International Symposium on Quality Electronic Design (ISQED), 2019 Publication	<1%
51	"Sign Language", Walter de Gruyter GmbH, 2012 Publication	<1%
52	"Information Technology and Intelligent Transportation Systems", Springer Science and Business Media LLC, 2017 Publication	<1%
53	Richard M. Duffy, Brendan D. Kelly. "Chapter 8 Incorporation of the United Nations' Convention on the Rights of Persons with Disabilities into Indian Law Through the Rights of Persons with Disabilities Act, 2016 and the Mental Healthcare Act, 2017", Springer Science and Business Media LLC, 2020 Publication	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		