

**INT 404- ARTIFICIAL INTELLIGENCE**

**Progress Report: Sign Recognition and Text Conversion**

**Submitted To: Sagar Pandey Sir**

**Submitted By:**

**Suveksha Chettri - 11811992**

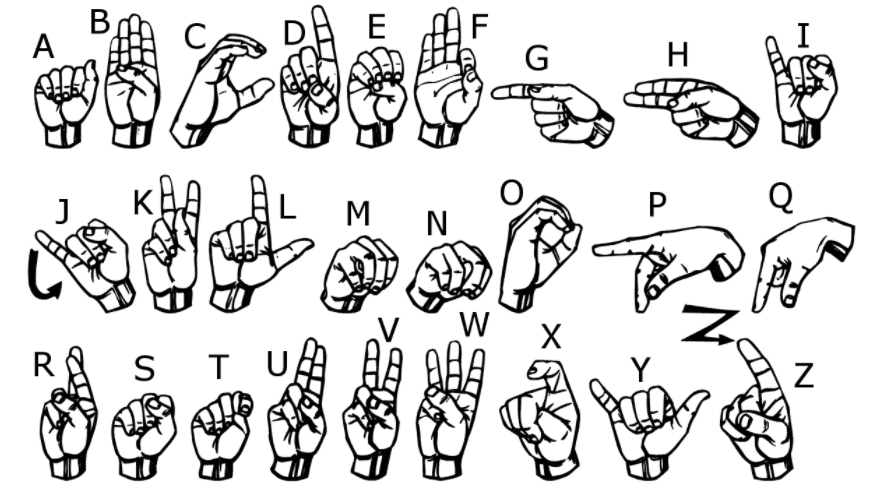
**Roll No.-17**

**Himanshu Singh - 11805048**

**Roll No.-11**

# 1.ABSTRACT

This project concerns with sign (language) recognition. Different hand gestures could be converted into their textual or conversational meaning. We have to train the AI with different hand gestures for that it can recognize them later on. Through this project we develop a software which can easily recognize sign language. Different gestures would be given for AI to remember with different angles for as near recognition as possible.



# INTRODUCTION

Sign language recognition is a collaborative research area which involves pattern matching, computer vision, natural language processing, and linguistics. Its objective is to build various methods and algorithms in order to identify already produced signs and to perceive their meaning

The project can further include many more hand gestures which have a meaning and can be made faster as to keep up with the person communicating with gestures so that sentences can be formed. This project can include or be divided into two major categories one being recognizing the gestures at real-time and the other being recognizing the gestures made by the person in a video or pictures.

# RELATED WORK

1. Garg P, Aggarwal N, Sofat S (2009) Vision based hand gesture recognition. World Acad Sci Eng Technol

2. Badhe PC, Kulkarni V (2015) Indian Sign Language translator using gesture recognition algorithm. In: Proceedings of IEEE international conference on computer graphics on vision and information security (CGVIS), Bhubaneshwar, India

3. Dour S, Kundargi M (2013) Design of ANFIS system for recognition of single hand and two hand signs for Indian Sign Language. Int J Appl Inf Syst

4. Amrutha CU, Davis N, Samrutha KS, Shilpa NS, Chunkath J (2016) application. Procedia Technol

# IMPLEMENTATION

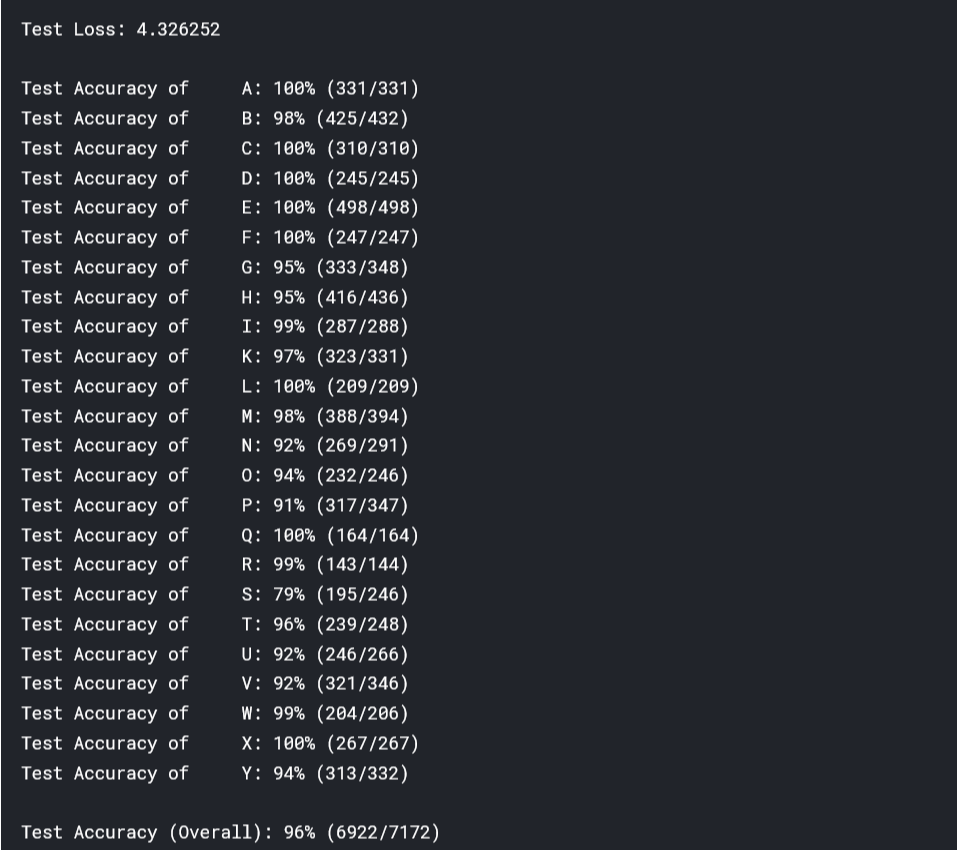
**Code for CNN:**

|  |
| --- |
| class Net(nn.Module): |
|  |  |
|  | def \_\_init\_\_(self): |
|  |  |
|  | super(Net,self).\_\_init\_\_() |
|  |  |
|  | #input depth , output depth , kernel size(filter)x |
|  |  |
|  | self.conv1=nn.Conv2d(1,32,kernel\_size=(3, 3),padding=(1, 1),stride=(1, 1)) |
|  |  |
|  | self.conv2=nn.Conv2d(32,32,kernel\_size=(3, 3),padding=(1, 1),stride=(1, 1)) |
|  |  |
|  | self.conv3=nn.Conv2d(32,64,kernel\_size=(3, 3),padding=(1, 1),stride=(1, 1)) |
|  |  |
|  | #padding for last conv layer |
|  | self.adapt = nn.AdaptiveMaxPool2d((3,3)) |
|  |  |
|  | #padding layer |
|  | self.pool=nn.MaxPool2d(2,2) |
|  |  |
|  | #dropout layer |
|  | self.drop=nn.Dropout(p=0.2) |
|  |  |
|  | #fc layers |
|  | self.fc1=nn.Linear(64\*3\*3,240) |
|  |  |
|  |  |
|  | self.fc2=nn.Linear(240,24) |
|  |  |
|  | self.softmax = nn.LogSoftmax(dim=1) |
|  |  |
|  | def forward(self,x): |
|  |  |
|  | x=self.pool(F.leaky\_relu(self.conv1(x))) |
|  |  |
|  | x=self.pool(F.leaky\_relu(self.conv2(x))) |
|  |  |
|  | x=self.adapt(F.leaky\_relu(self.conv3(x))) |
|  |  |
|  |  |
|  | #flatten Images |
|  | x = x.view(x.size(0), -1) |
|  |  |
|  | x=self.drop(x) |
|  |  |
|  | x=F.leaky\_relu(self.fc1(x)) |
|  |  |
|  |  |
|  | x=self.drop(x) |
|  |  |
|  | x=self.fc2(x) |
|  |  |
|  | return self.softmax(x) |
|  |  |

**Code for Processing the dataset:**

|  |
| --- |
| class DtProcessing(Dataset): |
|  |  |
|  | #initialise the class variables - transform, data, target |
|  | def \_\_init\_\_(self, data, target, transform=None): |
|  | self.transform = transform |
|  | self.data = data.reshape((-1,28,28)).astype(np.float32)[:,:,:,None] |
|  | # converting target to torch.LongTensor dtype |
|  | self.target = torch.from\_numpy(target).long() |
|  |  |
|  | #retrieve the X and y index value and return it |
|  | def \_\_getitem\_\_(self, index): |
|  | return self.transform(self.data[index]), self.target[index] |
|  |  |
|  | #returns the length of the data |
|  | def \_\_len\_\_(self): |
|  | return len(list(self.data)) |
|  |  |
|  |  |
|  | #divide train set into train and validation set |
|  |  |
|  | from sklearn.model\_selection import train\_test\_split |
|  |  |
|  | X\_train, X\_valid, y\_train, y\_valid = train\_test\_split(train\_data, new\_train\_labels, test\_size = .2, random\_state = 42) |
|  |  |
|  |  |
|  | dset\_train = DtProcessing(X\_train, y\_train, transform=data\_transforms['train']) |
|  |  |
|  | train\_loader = torch.utils.data.DataLoader(dset\_train, batch\_size=20, |
|  | shuffle=True, num\_workers=0) |
|  |  |
|  | dset\_valid = DtProcessing(X\_valid, y\_valid, transform=data\_transforms['valid']) |
|  |  |
|  | valid\_loader = torch.utils.data.DataLoader(dset\_valid, batch\_size=20, |
|  | shuffle=True, num\_workers=0) |
|  |  |
|  |  |
|  | dset\_test = DtProcessing(test\_data, new\_test\_labels, transform=data\_transforms['valid']) |
|  |  |
|  | test\_loader =torch.utils.data.DataLoader(dset\_test, batch\_size=32, shuffle=True) |
|  |  |

# RESULTS

****

# LIBRARIES USED

**PyTorch: PyTorch** is an [open source](https://en.wikipedia.org/wiki/Open_source) [machine learning](https://en.wikipedia.org/wiki/Machine_learning) [library](https://en.wikipedia.org/wiki/Library_(computing)) based on the [Torch](https://en.wikipedia.org/wiki/Torch_(machine_learning)) library, used for applications such as [computer vision](https://en.wikipedia.org/wiki/Computer_vision) and [natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing). It is primarily developed by [Facebook](https://en.wikipedia.org/wiki/Facebook)'s AI Research lab (FAIR).

**Torch Vision**: The torch vision package consists of popular datasets, model architectures, and common image transformations for computer vision.

**Other**

Python 3

**CNN**: Introduction. A Convolutional Neural Network (**CNN**), sometimes referred to as a ConvNet, is the most well-known image recognition and classification algorithm. This is used in this project to capture the hand gestures.

# REFERENCES

[www.towardsdatascience.com](http://www.towardsdatascience.com/)

Guide To understanding Convolutional Neural Network – Adesh Pandey