CSE3505 – FOUNDATIONS OF DATA ANALYTICS

Project Report

TITLE:- SIGN LANGUAGE GENERATION

By

19MIA1004 HARINI GOKULRAM NAIDU

19MIA1006 SHIVANI GOKULRAM NAIDU

19MIA1008 P. SUBHASHRI

M. Tech (Intg) Computer Science Engineering with specialisation in Business Analytics

Submitted to

Dr. R. PRIYADARSHINI

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

November 2022

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report entitled “Sign Language generation” is a bonafide work of HARINI GOKULRAM NAIDU(19MIA1004), SHIVANI GOKULRAM NAIDU(19MIA1006), P.SUBHASHRI(19MIA1008) who carried out the J-component under my supervision and guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Dr. R. Priyadarshini,

VIT, Chennai

Contents

ABSTRACT

Speech impairment is a disability which affects an individual's ability to communicate using speech and hearing. People who are affected by this use other media of communication such as sign language. Although sign language is ubiquitous in recent times, there remains a challenge for non-sign language speakers to communicate with sign language speakers or signers. With recent advances in deep learning and computer vision there has been promising progress in the fields of motion and gesture recognition using deep learning and computer vision-based techniques. The focus of this work is to create a vision-based application which offers sign language translation to text thus aiding communication between signers and non-signers. We use CNN (Convolutional Neural Network), VGG16 and RESNET50 models. The dataset used is the American Sign Language Dataset. Later on we are building a sign language generator from audio.

INTRODUCTION

Language is undoubtedly essential to human interaction and has existed since human civilisation began. It is a medium humans use to communicate to express themselves and understand notions of the real world. Without it, no books, no cell phones and definitely not any word I am writing would have any meaning. It is so deeply embedded in our everyday routine that we often take it for granted and don't realise its importance. Sadly, in the fast-changing society we live in, people with hearing impairment are usually forgotten and left out. They have to struggle to bring up their ideas, voice out their opinions and express themselves to people who are different to them. Sign language, although being a medium of communication to deaf people, still have no meaning when conveyed to a non-sign language user. Hence, broadening the communication gap. To prevent this from happening, we are putting forward a sign language recognition system. It will be an ultimate tool for people with hearing disability to communicate their thoughts as well as a very good interpretation for non-sign language user to understand what the latter is saying. Many countries have their own standard and interpretation of sign gestures. For instance, an alphabet in Korean sign language will not mean the same thing as in Indian sign language. While this highlights diversity, it also pinpoints the complexity of sign languages. Deep learning must be well versed with the gestures so that we can get a decent accuracy. In our proposed system, American Sign Language is used to create our datasets.

LITERATURE SURVEY

1) STATISTICAL AND SPATIO-TEMPORAL HAND GESTURE FEATURES FOR SIGN LANGUAGE RECOGNITION USING THE LEAP MOTION SENSOR

Jordan J.Bird

The experiments in this work consider the problem of SL gesture recognition regarding how dynamic gestures change during their delivery, and this study aims to explore how single types of features as well as mixed features affect the classification ability of a machine learning model. 18 common gestures recorded via a Leap Motion Controller sensor provide a complex classification problem.

DRAWBACKS

FIrstly the number of chosen raw features prior to extraction was set based on an F-score cutoff point, due to which the quality of the extracted features is less. On the point of feature selection, this study focused on F-scores for comparability, but other methods of selection must also be explored and compared.

2) TOWARDS ZERO-SHOT SIGN LANGUAGE RECOGNITION

Yunus Can Bilge, Ramazan Gokberk Cinbis, and Nazli Ikizler-Cinbis

This paper tackles the problem of zero-shot sign language recognition (ZSSLR), where the goal is to leverage models learned over the seen sign classes to recognize the instances of unseen sign classes. In this context, readily available textual sign descriptions and attributes collected from sign language dictionaries are utilized as semantic class representations for knowledge transfer.

DRAWBACKS

First, some of the differences across sign descriptions are subtle, both visually and textually. Second, many dialects exists; even the same sign can be expressed in many different forms.

3) SIGN LANGUAGE RECOGNITION SYSTEM USING TENSORFLOW OBJECT DETECTION API

Sharvani Srivastava, Amisha Gangwar, Richa Mishra, Sudhakar Singh

In this paper, a method is proposed to create an Indian Sign Language dataset using a webcam and then using transfer learning, train a TensorFlow model to create a real-time Sign Language Recognition system. The system achieves a good level of accuracy even with a limited size dataset.

DRAWBACKS

Though the system has achieved a high average confidence rate, the dataset it has been trained on is small in size and limited. The dataset is small hence the system cannot recognize more gestures.

4) ALL YOU NEED IN SIGN LANGUAGE PRODUCTION

Razieh Rastgoo, Kourosh Kiani, Sergio Escalera ,Vassilis Athitsos ,Mohammad Sabokrou

This paper presents the fundamental components of a bi-directional sign language translation system, discussing the main challenges in this area. Also, the backbone architectures and methods in SLP are briefly introduced and the proposed taxonomy on SLP is presented.

DRAWBACKS

The possibility of high-resolution and photo-realistic continuous sign language videos. Most of the proposed models in SLP can only generate low-resolution sign samples. Conditioning on human keypoints extracted from training data can decrease the parameter complexity of the model and assist to produce a high-resolution video sign.

5) END-TO-END SIGN LANGUAGE TRANSLATION VIA MULTITASK LEARNING

This paper extends the ordinary Transformer decoder with two channels to support multitasking, where each channel is devoted to solve a particular problem. To control the memory footprint of our model, channels are designed to share most of their parameters among each other.

DRAWBACKS

As this approach is both model and task agnostic, this approach could have also been extended to other language understanding (NLU) tasks using various deep learning architectures.

6) USING MOTION HISTORY IMAGES WITH 3D CONVOLUTIONAL NETWORKS IN ISOLATED SIGN LANGUAGE RECOGNITION

Ozge Mercanoglu Sincan, Hacer Yalim Keles

In this paper, they have proposed an isolated sign language recognition model based on a model trained using Motion History Images (MHI) that are generated from RGB video frames. RGB-MHI images represent spatio-temporal summary of each sign video effectively in a single RGB image. Two different approaches have been proposed using this RGB-MHI model. In the first approach, the RGB-MHI model as a motion-based spatial attention module integrated into a 3D-CNN architecture is being used. In the second approach, the RGB-MHI model features directly with the features of a 3D-CNN model using a late fusion technique.

DRAWBACKS

Haven't investigated the effectiveness of the proposed models in the continuous SLR domain.

DATASET

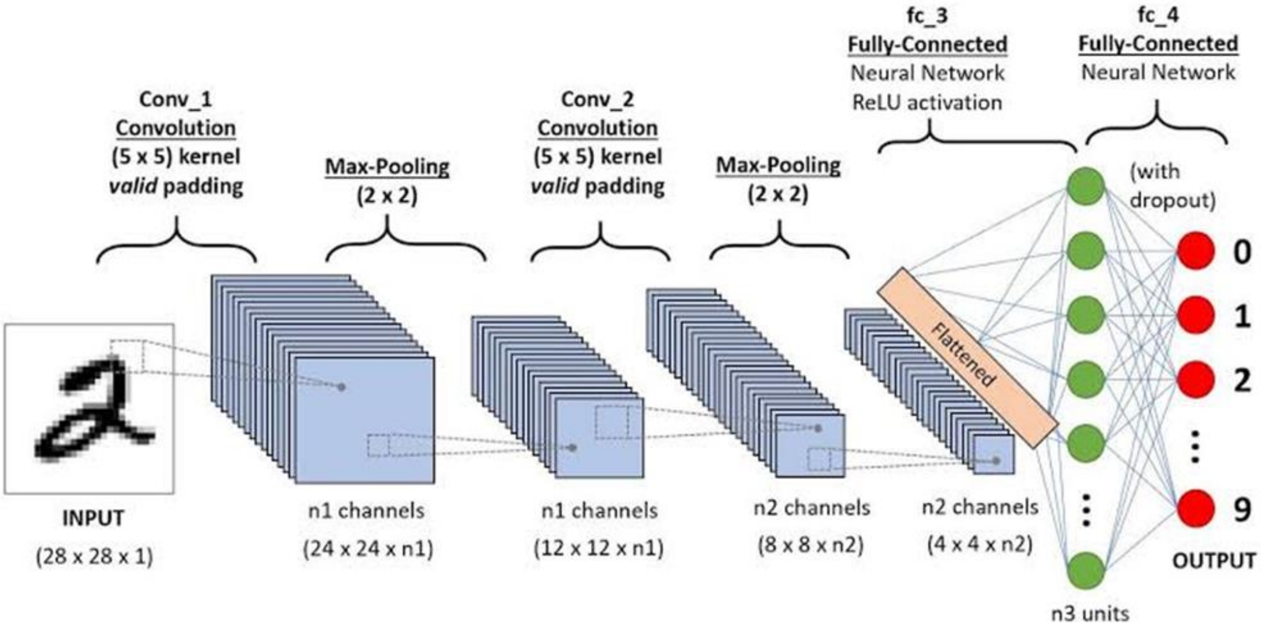
The Dataset is taken from a Kaggle Repository. It's the American Sign Language dataset. Used for the Model Training purpose and used to build the model. The dataset contains 1400 images for each alphabet, digits and some custom words like question mark sign, space etc. There are two folders training and testing. The test dataset is customized. In the training set, there are 56000 images and in test set there are 8000 images.

PROPOSED METHODOLOGY

ALGORITHMS

CNN

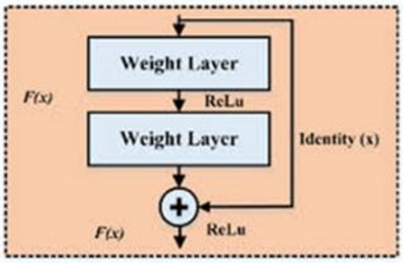
In deep learning, a convolutional neural network (CNN/ConvNet) is a class of deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. Now in mathematics convolution is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other.



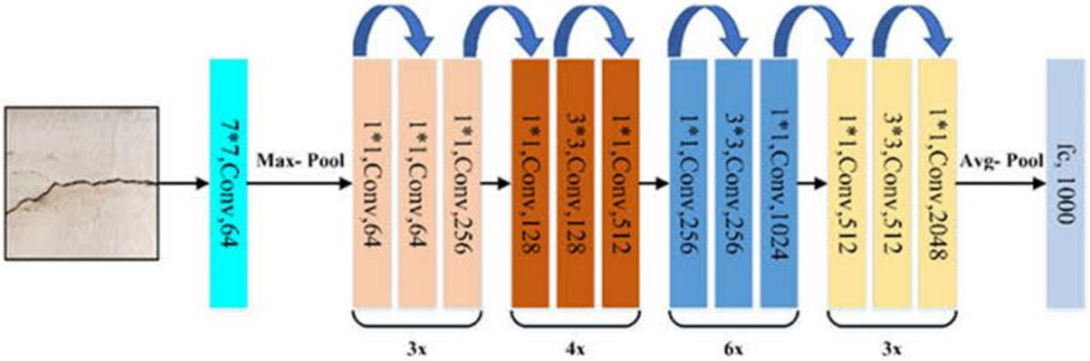
RESNET 50

ResNet stands for Residual Network and is a specific type of convolutional neural network (CNN) introduced in the 2015 paper "Deep Residual Learning for Image Recognition".ResNet-50 is a 50-layer convolutional neural network (48 convolutional layers, one MaxPool layer, and one average pool layer).

Residual neural networks are a type of artificial neural network (ANN) that forms networks by stacking residual blocks.



Residual Learning Block



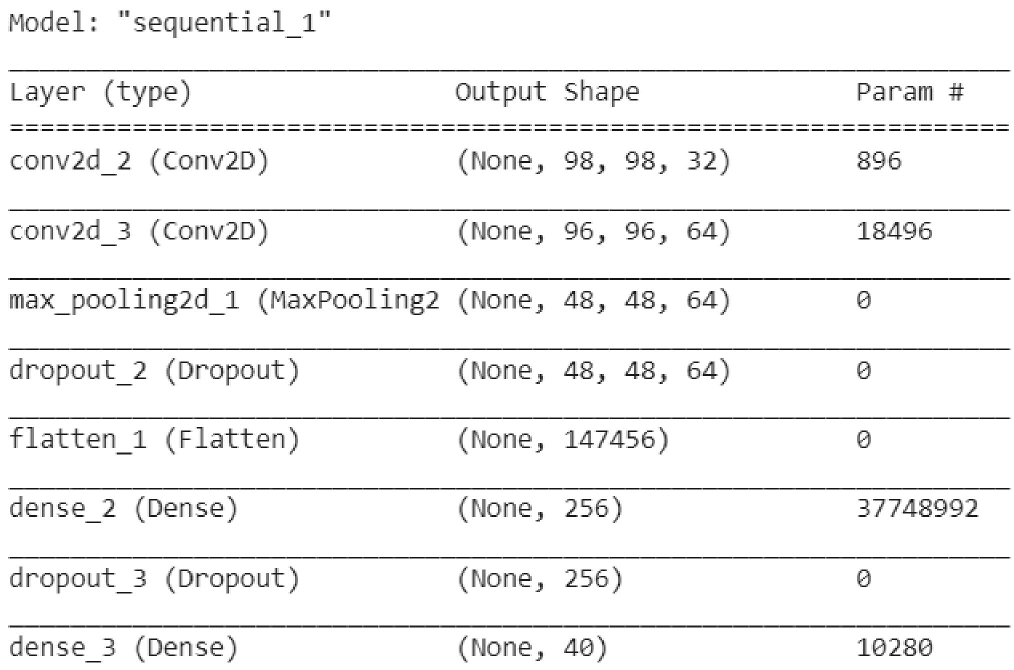
VGG 16

A convolutional neural network is also known as a ConvNet, which is a kind of artificial neural network. A convolutional neural network has an input layer, an output layer, and various hidden layers. VGG16 is a type of CNN (Convolutional Neural Network) that is considered to be one of the best computer vision models to date. The creators of this model evaluated the networks and increased the depth using an architecture with very small (3 × 3) convolution filters, which showed a significant improvement on the prior-art configurations. They pushed the depth to 16–19 weight layers making it approx — 138 trainable parameters.

IMPLEMENTATION AND RESULTS

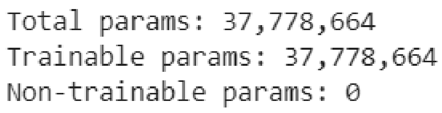
❖ CNN

MODEL



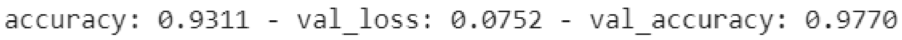
```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_2 (Conv2D)            (None, 98, 98, 32)        896       
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 96, 96, 64)        18496     
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 48, 48, 64)        0         
_________________________________________________________________
dropout_2 (Dropout)          (None, 48, 48, 64)        0         
_________________________________________________________________
flatten_1 (Flatten)          (None, 147456)            0         
_________________________________________________________________
dense_2 (Dense)              (None, 256)               37748992  
_________________________________________________________________
dropout_3 (Dropout)          (None, 256)               0         
_________________________________________________________________
dense_3 (Dense)              (None, 40)                10280     
```

PARAMETERS

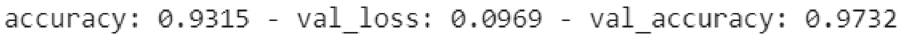


```
Total params: 37,778,664
Trainable params: 37,778,664
Non-trainable params: 0
```

ACCURACY

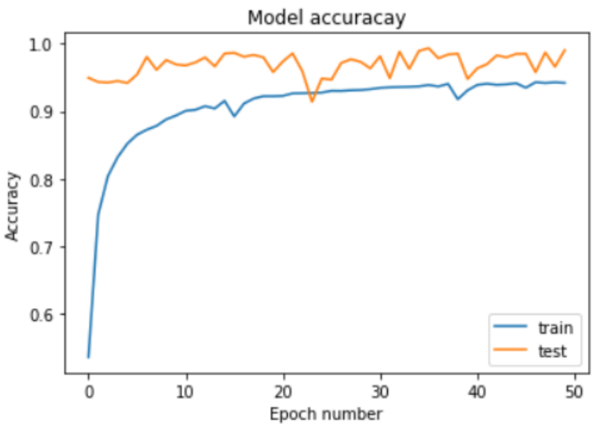


```
accuracy: 0.9311 - val_loss: 0.0752 - val_accuracy: 0.9770
```



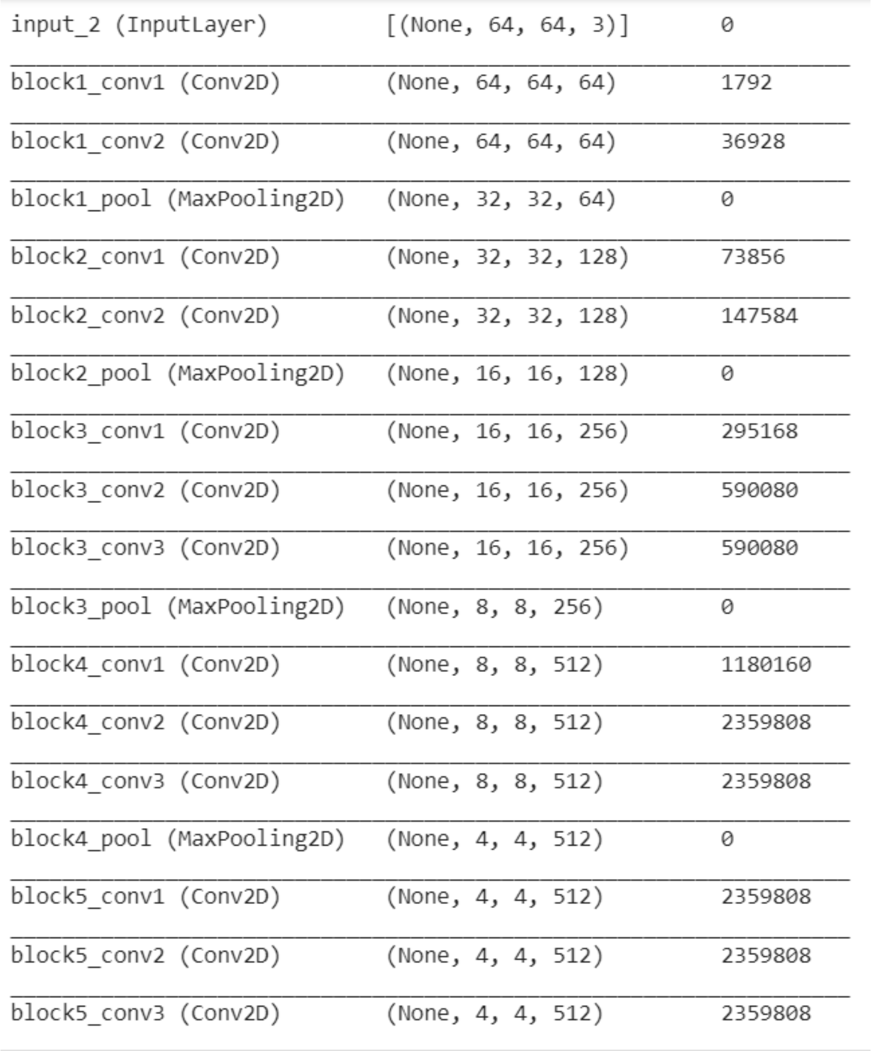
```
accuracy: 0.9315 - val_loss: 0.0969 - val_accuracy: 0.9732
```

PLOT



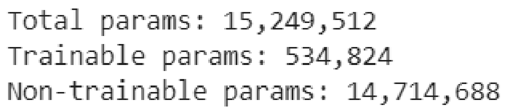
❖ **VGG 16**

MODEL



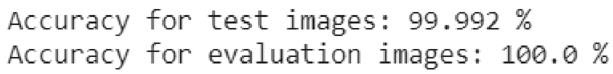
```
input_2 (InputLayer)         [(None, 64, 64, 3)]       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 64, 64, 64)        1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 64, 64, 64)        36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 32, 32, 64)        0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 32, 32, 128)       73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 32, 32, 128)       147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 16, 16, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 16, 16, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 16, 16, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 16, 16, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 8, 8, 256)         0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 8, 8, 512)         1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 8, 8, 512)         2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 8, 8, 512)         2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 4, 4, 512)         0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 4, 4, 512)         2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 4, 4, 512)         2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 4, 4, 512)         2359808
```

PARAMETERS



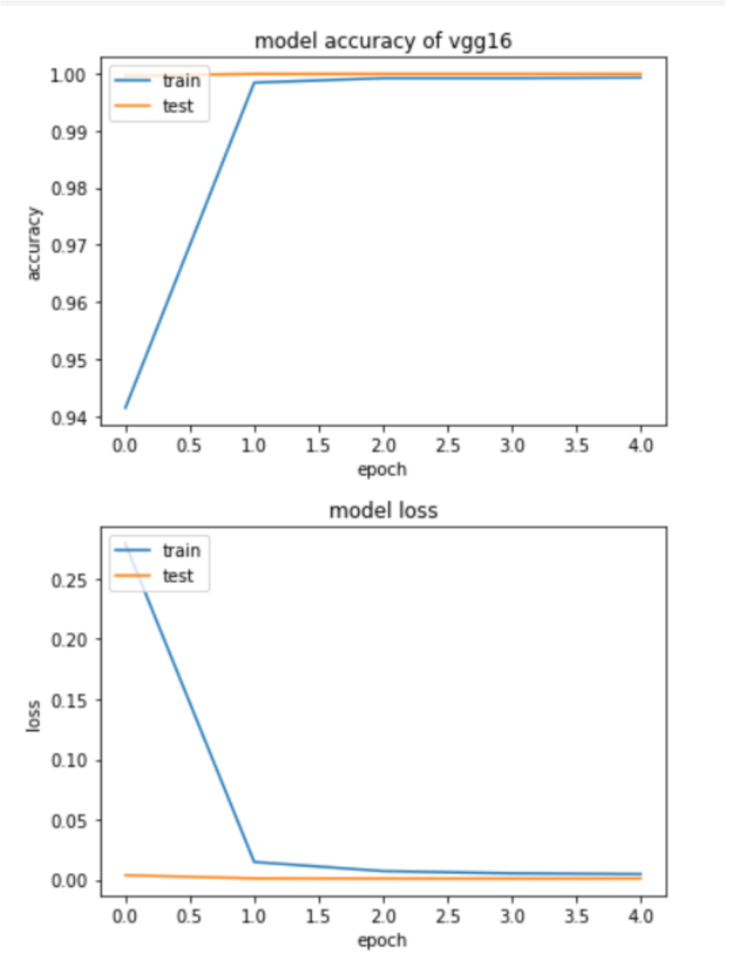
```
Total params: 15,249,512
Trainable params: 534,824
Non-trainable params: 14,714,688
```

ACCURACY



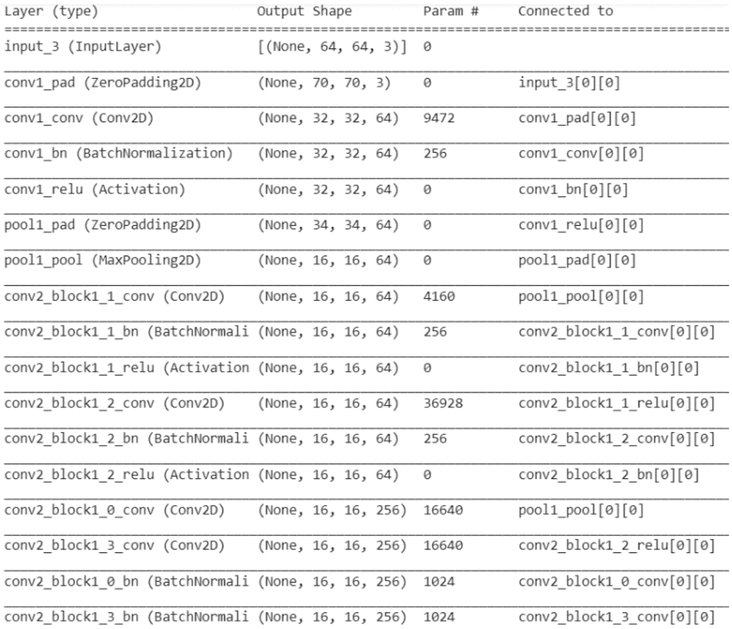
```
Accuracy for test images: 99.992 %
Accuracy for evaluation images: 100.0 %
```

PLOTS



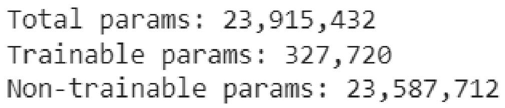
- ❖ **RESNET 50**

MODEL



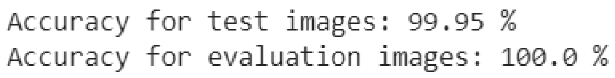
```
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_3 (InputLayer)            [(None, 64, 64, 3)]  0
__________________________________________________________________________________________________
conv1_pad (ZeroPadding2D)       (None, 70, 70, 3)    0           input_3[0][0]
__________________________________________________________________________________________________
conv1_conv (Conv2D)             (None, 32, 32, 64)   9472        conv1_pad[0][0]
__________________________________________________________________________________________________
conv1_bn (BatchNormalization)   (None, 32, 32, 64)   256         conv1_conv[0][0]
__________________________________________________________________________________________________
conv1_relu (Activation)         (None, 32, 32, 64)   0           conv1_bn[0][0]
__________________________________________________________________________________________________
pool1_pad (ZeroPadding2D)       (None, 34, 34, 64)   0           conv1_relu[0][0]
__________________________________________________________________________________________________
pool1_pool (MaxPooling2D)       (None, 16, 16, 64)   0           pool1_pad[0][0]
__________________________________________________________________________________________________
conv2_block1_1_conv (Conv2D)    (None, 16, 16, 64)   4160        pool1_pool[0][0]
__________________________________________________________________________________________________
conv2_block1_1_bn (BatchNormali (None, 16, 16, 64)   256         conv2_block1_1_conv[0][0]
__________________________________________________________________________________________________
conv2_block1_1_relu (Activation (None, 16, 16, 64)   0           conv2_block1_1_bn[0][0]
__________________________________________________________________________________________________
conv2_block1_2_conv (Conv2D)    (None, 16, 16, 64)   36928       conv2_block1_1_relu[0][0]
__________________________________________________________________________________________________
conv2_block1_2_bn (BatchNormali (None, 16, 16, 64)   256         conv2_block1_2_conv[0][0]
__________________________________________________________________________________________________
conv2_block1_2_relu (Activation (None, 16, 16, 64)   0           conv2_block1_2_bn[0][0]
__________________________________________________________________________________________________
conv2_block1_0_conv (Conv2D)    (None, 16, 16, 256)  16640       pool1_pool[0][0]
__________________________________________________________________________________________________
conv2_block1_3_conv (Conv2D)    (None, 16, 16, 256)  16640       conv2_block1_2_relu[0][0]
__________________________________________________________________________________________________
conv2_block1_0_bn (BatchNormali (None, 16, 16, 256)  1024        conv2_block1_0_conv[0][0]
__________________________________________________________________________________________________
conv2_block1_3_bn (BatchNormali (None, 16, 16, 256)  1024        conv2_block1_3_conv[0][0]
```

PARAMETERS



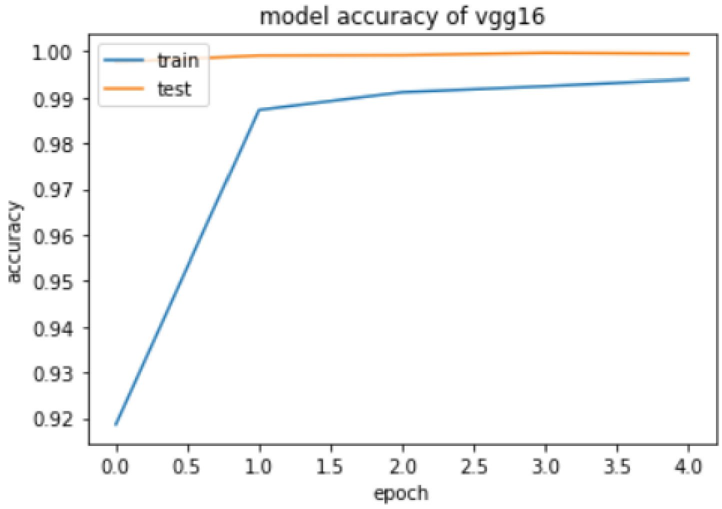
```
Total params: 23,915,432
Trainable params: 327,720
Non-trainable params: 23,587,712
```

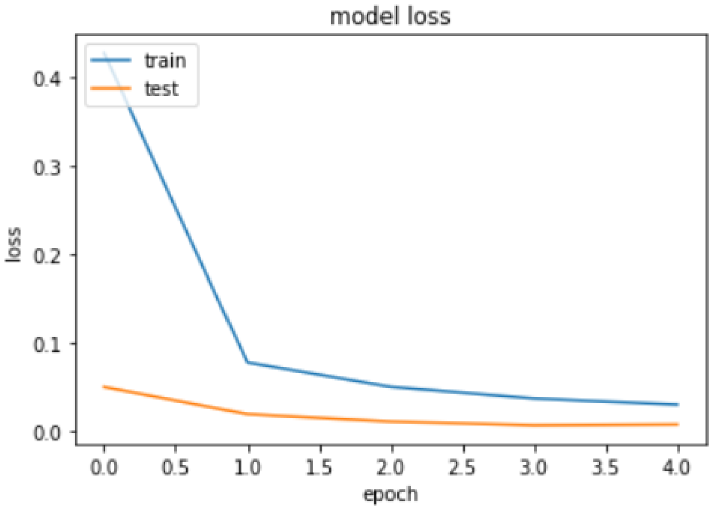
ACCURACY



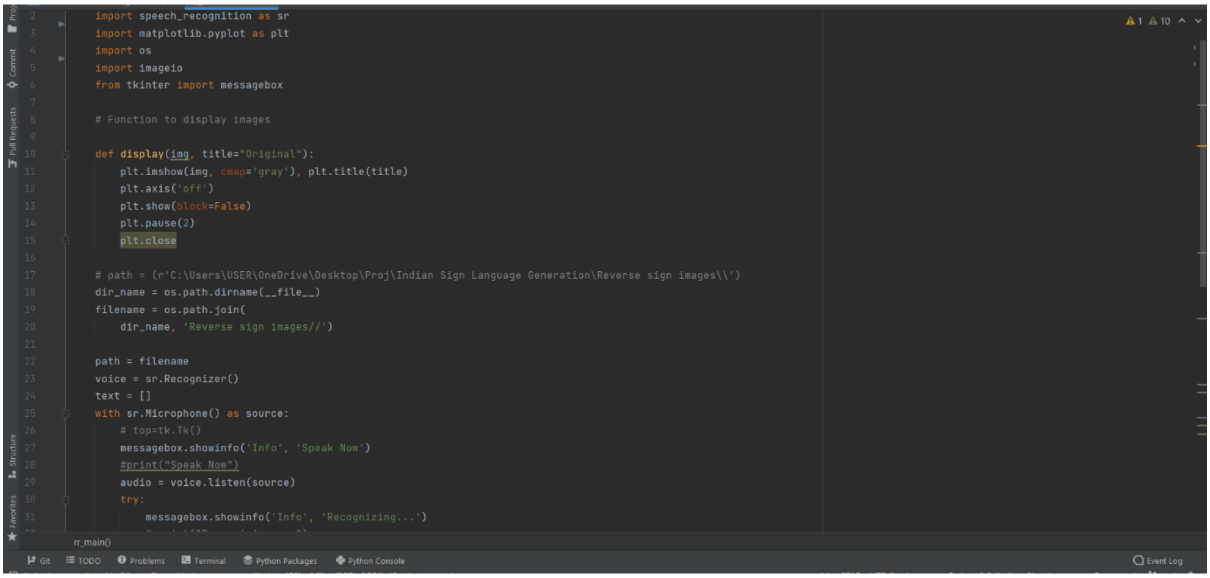
```
Accuracy for test images: 99.95 %
Accuracy for evaluation images: 100.0 %
```

PLOTS


model accuracy of vgg16
train
test
accuracy
1.00
0.99
0.98
0.97
0.96
0.95
0.94
0.93
0.92
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
epoch


model loss
train
test
loss
0.4
0.3
0.2
0.1
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
epoch

SPEECH TO TEXT GENERATOR



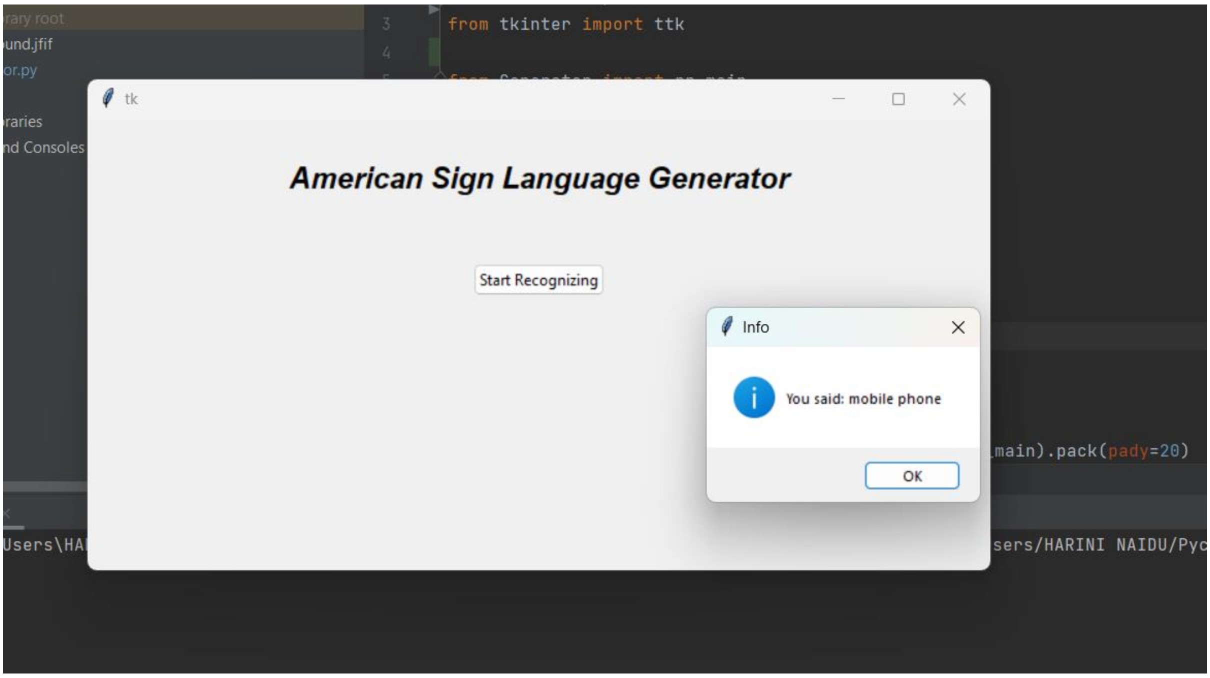
```
import speech_recognition as sr
import matplotlib.pyplot as plt
import os
import imageio
from tkinter import messagebox

# Function to display images

def display(img, title="Original"):
    plt.imshow(img, cmap='gray'), plt.title(title)
    plt.axis('off')
    plt.show(block=False)
    plt.pause(2)
    plt.close

# path = (r'C:\Users\USER\OneDrive\Desktop\Proj\Indian Sign Language Generation\Reverse sign images\\')
dir_name = os.path.dirname(__file__)
filename = os.path.join(
    dir_name, 'Reverse sign images//')

path = filename
voice = sr.Recognizer()
text = []
with sr.Microphone() as source:
    # top=tk.Tk()
    messagebox.showinfo('Info', 'Speak Now')
    #print("Speak Now")
    audio = voice.listen(source)
    try:
        messagebox.showinfo('Info', 'Recognizing...')
```



```
from tkinter import ttk
```

American Sign Language Generator

Info

You said: mobile phone

CONCLUSION

Many breakthroughs have been made in the field of artificial intelligence, machine learning and computer vision. They have immensely contributed in how we perceive things around us and improve the way in which we apply their techniques in our everyday lives. We have used CNN, VGG16 and RESNET50 to classify the alphabets and also generated a speech to text GUI model. We arrived at an accuracy of 98% approx..

REFERENCES

- **Kapur, R.: The Types of Communication. MIJ. 6, (2020).**
- **2Suharjito, Anderson, R., Wiryana, F., Ariesta, M.C., Kusuma, G.P.: Sign Language Recognition Application Systems for Deaf-Mute People: A Review Based on InputProcess-Output. Procedia Comput. Sci. 116, 441–448 (2017). https://doi.org/10.1016/J.PROCS.2017.10.028.**
- **https://kccemsr.edu.in/public/files/technovision/1/SIGN%20LANGUAGE%20RECOGNITION%20USING%20NEURAL%20NETWORK.pdf**
- **Bragg, D., Koller, O., Bellard, M., Berke, L., Boudreault, P., Braffort, A., Caselli, N., Huenerfauth, M., Kacorri, H., Verhoef, T., Vogler, C., Morris, M.R.: Sign Language Recognition, Generation, and Translation: An Interdisciplinary Perspective. 21st Int. ACM SIGACCESS Conf. Comput. Access. (2019). https://doi.org/10.1145/3308561.**
- **https://theses.eurasip.org/media/theses/documents/aran-oya-vision-based-sign-language-recognition-modeling-and-recognizing-isolated-signs-with-manual-and-non-manual-components.pdf**
- **Rosero-Montalvo, P.D., Godoy-Trujillo, P., Flores-Bosmediano, E., Carrascal-Garcia, J., 12 Otero-Potosi, S., Benitez-Pereira, H., Peluffo-Ordonez, D.H.: Sign Language Rec**
- **https://www.e3s-conferences.org/articles/e3sconf/abs/2022/18/e3sconf_icies2022_01065/e3sconf_icies2022_01065.html**