1. **Abstract-**

Hand gestures are form nonverbal communication that can be used several fields such as communication between deaf-mute people, robot control, human-computer interaction and medical applications. The hand sign can be classified under many headings, such as posture and gesture, as well dynamic and static. Human-Computer Interaction (HMI) is useful in sterile environments such as operating rooms (OR) where surgeons need to interact with images from scanners of organs on screens. Contamination issues may happen if the surgeon must touch a keyboard or the mouse. In order to reduce contamination and improve the interactions with the images without asking another team member, the process is carried on. The above-mentioned problem is what we worked on. In order to detect the signs, deep learning methods have been programmed using a pretrained Convolutional Neural Network. For such development this model will be working with Convolution Neural Network, with different classification methods. Rotation, Information, Zoom-In, Zoom-Out images on screen according to the recognised sign. Future work will include tests in real situations in an operating room to obtain feedback from doctors to improving the system.

1. **Introduction**

Hand gesture recognition is one of the most advanced domains in which computer vision and artificial intelligence has helped to improve communication. Touchless Human-Machine Interface (HMI) is a field with applications in robotics, computer gaming and sign-language interpretation. Moreover, touchless HMI is very useful in sterile environments such as doctors or surgeons need to interact with reports with computers without introducing contamination. Most of the time, buttons, or touch screens are wrapped in a plastic and the surgeons need to change their gloves each time they have to use the computers. It is quite common for surgeons to ask colleagues or nurses, who are in another room to interact with the computers for moving images. The progress of the gesture recognition systems plays a vital role in the development of computer and human interaction. The main aim of this real-time hand gesture touchless recognition application is to classify and recognize the gestures and then perform the particular task /movement assigned to them using Deep Learning method. This is possible with the help of touchless gesture recognition application where you can access the images without coming into the contact of the screen by showing the gestures interfaced in your machine.

1. **Literature Review**

Humans tend to naturally use hand gestures in their communication process to clarify their intentions, which gives computers the ability of capturing and interpreting hand gestures, and executing commands afterwards.  The aim of this study is to perform identifying the most prominent techniques, applications and challenges in contactless hand gesture exploration of medical images.

**3.1.** [**Vision based static hand gesture recognition techniques**](https://ieeexplore.ieee.org/abstract/document/8286451/)**:**

**Author: A. Sharma, A. Khandelwal, K. Kaur**

**Dated: 08 February 2018**

* This is the study of how people interact with machines or specially computers in an efficient manner, one such approach is static hand gesture recognition.
* The main aim of gesture recognition system is to detect the specific gesture of hand and use the information for a variety of applications. Current work deals with explaining the steps involved in vision-based automatic hand gesture recognition, along with elaborating some of the pre-processing, feature extraction and classification techniques.

**3.2. Hand-Gesture Recognition based on Computer Vision:**

**Author: Munir Oudah, Ali Al-Naji and Javaan Chahl**

**Dated: 23 July 2020**

* Development of promising and cost-effective techniques are called camera vision-based sensor technologies. With the evolution of open-source software libraries, it is easier than ever to detect hand gestures that can be used under a wide range of applications like clinical operations, sign language, robot control, virtual environments, home automation, personal computer and tablet, gaming. These techniques essentially involve replacement of the instrumented glove with a camera. Different types of cameras are used for this purpose, such as RGB camera, time of flight (TOF) camera, thermal cameras or night vision cameras.
* Algorithms have been developed based on computer vision methods to detect hands using these different types of cameras. The algorithms attempt to segment and detect hand features such as skin colour, appearance, motion, skeleton, depth, 3D model, deep learn detection and more.

**3.3. Real-Time Hand Gesture Interface for Browsing Medical Images**

**Author: Juan Wachs, Helman Stern, Yael Edan, Michael Gillam, Craig Feied, Mark Smith, Jon Handler**

**Dated: 3 March 2007**

* A web-camera placed above a screen captures a sequence of images of the hand. The hand is tracked by a tracking module which segments the hand from the background using colour and motion cues. This is followed by black/white thresholding.
* The location of the hand in each image is represented by the 2D coordinates of its centroid. This spatio-temporal data is mapped into a ‘flick gesture’.
* A flick gesture is the rapid movement of the hand from a neutral position to a specific direction. The direction of the ‘flick’ is used to navigate through a visual image data browser. Other actions/commands such as: zoom, rotate are recognized by extracting features from the spatio-temporal data of the gestures. With these actions/commands doctors can bring up X-rays images, select a patient record from the database or move objects and windows in the screen. A two-layer architecture is used.
* Real time feedback and operation - during surgery the system should be fast and enable the surgeon to obtain visual feedback of the evoked gestures.

**3.4. A Touchless system for image visualization during surgery:**

2018 Jul

* This work aims to assess during tests in laboratory and clinical settings a Surgery Touchless System. This system allows clinicians to interact with medical images by using two different approaches: a gesture recognition and a voice recognition-based system.
* These two methods are based on the use of a Microsoft Kinect and of a selective microphone, respectively. The STS allows navigating in a specifically designed interface, to perform several tasks, among others, to manipulate biomedical images.

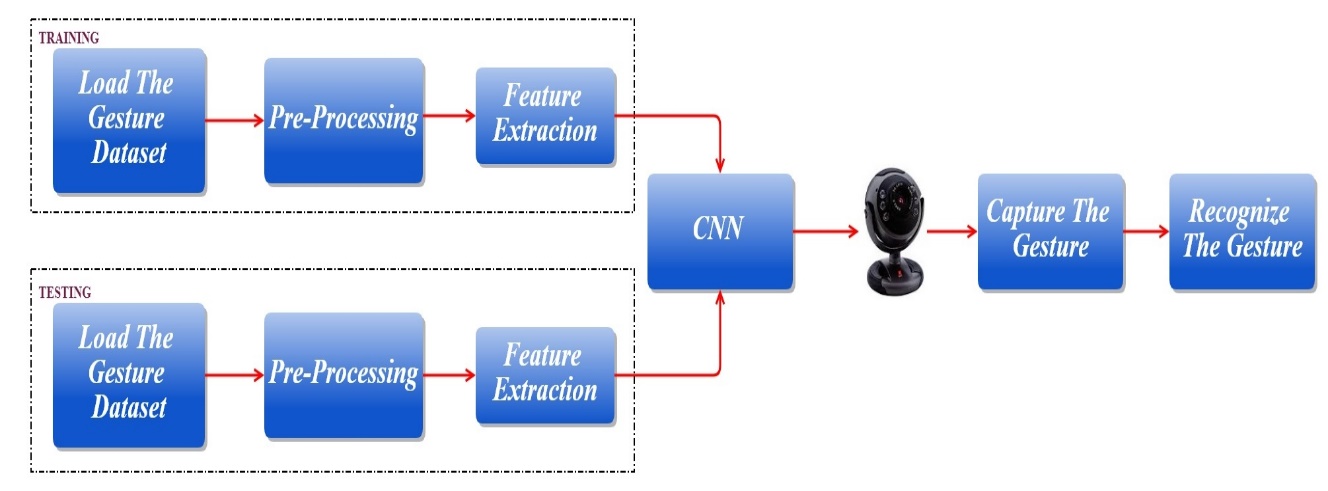
**3.5. Hand Gesture Recognition using Deep Learning Neural Networks**

**Author: Norah Meshari Alnaim**

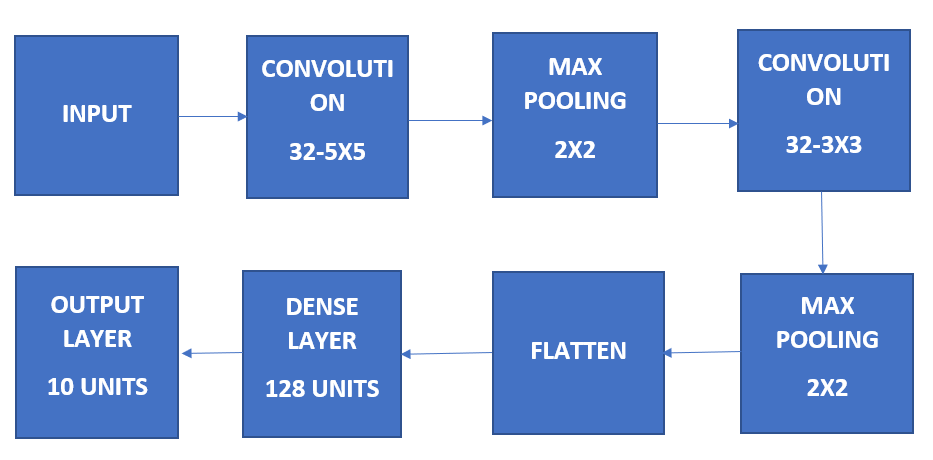
**Dated:** **December 2019**

* The aim of the research is to develop a system for 2D, and 3D hand gesture recognition using any type of camera, background, illuminations or position of hand, by finding the most appropriate algorithms to implement the system and test the validation of system.
* This system helps individuals with special needs and people who have experienced stroke to communicate accurately. Using WT and EMD algorithms for feature extraction and AI for classification provides different results while CNN provides an accurate result. Determine the type of gestures, record them using different cameras, such as mobile cameras and a holoscopic imaging system camera, as well as apply them into a pre-processing phase before analysing them.
* Develop a classification system using ANN and CNN classifiers.
* Sign language is one of the common examples for a hand gesture system. It is defined as a linguistic system based on hand motions besides other motions. For instance, most hearing-impaired people around the world use universal sign language. Sign language contains three fundamental parts: word level sign vocabulary, non-manual features and finger spelling. One of best methods to communicate with hearing-impaired people is sign language.

1. **Block Diagram:**



**4.1 Block diagram of the model:**

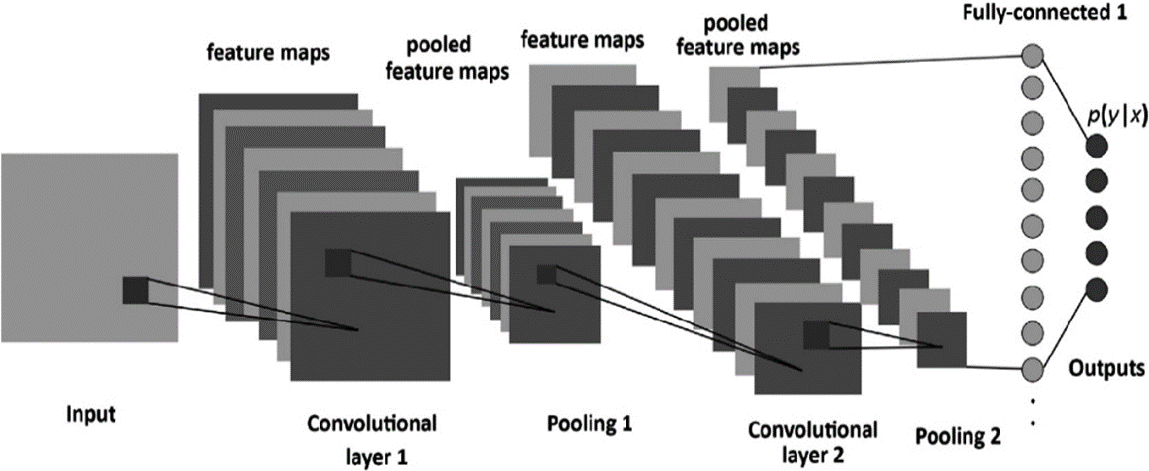


* Initially the dataset is loaded into the network for training. Pre-processing is done before the feature is extracted.  The training is done in Convolutional neural network.  After training an input image is given by capturing from a webcam. The given input image is tested for recognizing the gesture. A confusion matrix is produced accordingly to the produced output with its mean accuracy.
* A ConvNet is a popular machine learning algorithm. Which is one of the techniques of Deep learning and is a learning model used to execute classification tasks through images. CNNs specifically give better results for identifying patterns in an image, which leads to recognizing of hand gesture. The advantage of CNN is it don’t require any feature extraction to train the model. CNN is invariant to the scaling and rotation.
* The first step is to gather the data. The data must be labelled. If the dataset is not labelled, this can be time consuming as you would have to manually create new labels for each category of images. Another method is to create new labels and only move pictures into their proper labels, and create a classifier like the one we will and have that machine classify the images.
* The data set that is required is then pre-processed as per the requirement. The feature extraction takes place i.e., the main features which are shown in the dataset images are extracted for the classification to happen. Then these images are given to CNN i.e., Convolutional Neural Network. And this Convolutional Neural Network has many functioning layers and each layer in it has its own specifications to perform.
* Now that we have our datasets stored safely in our computer, let’s make sure we have a training data set, a validation data set, and a testing data set.
* Training data set would contain 85–90% of the total labelled data. This data would be used to train our machine about the different types of images we have. Validation data set would contain 5–10% of the total labelled data. This will test how well our machine performs against known labelled data. This testing data will be used to test how well our machine can classify data it has never seen.

1. **HAND GESTURE DATASET COLLECED:**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **ZOOM IN** | **ZOOM OUT** | **INFORMATION** | **ROTATION** |

1. **Convolutional Neural Network Architecture:**

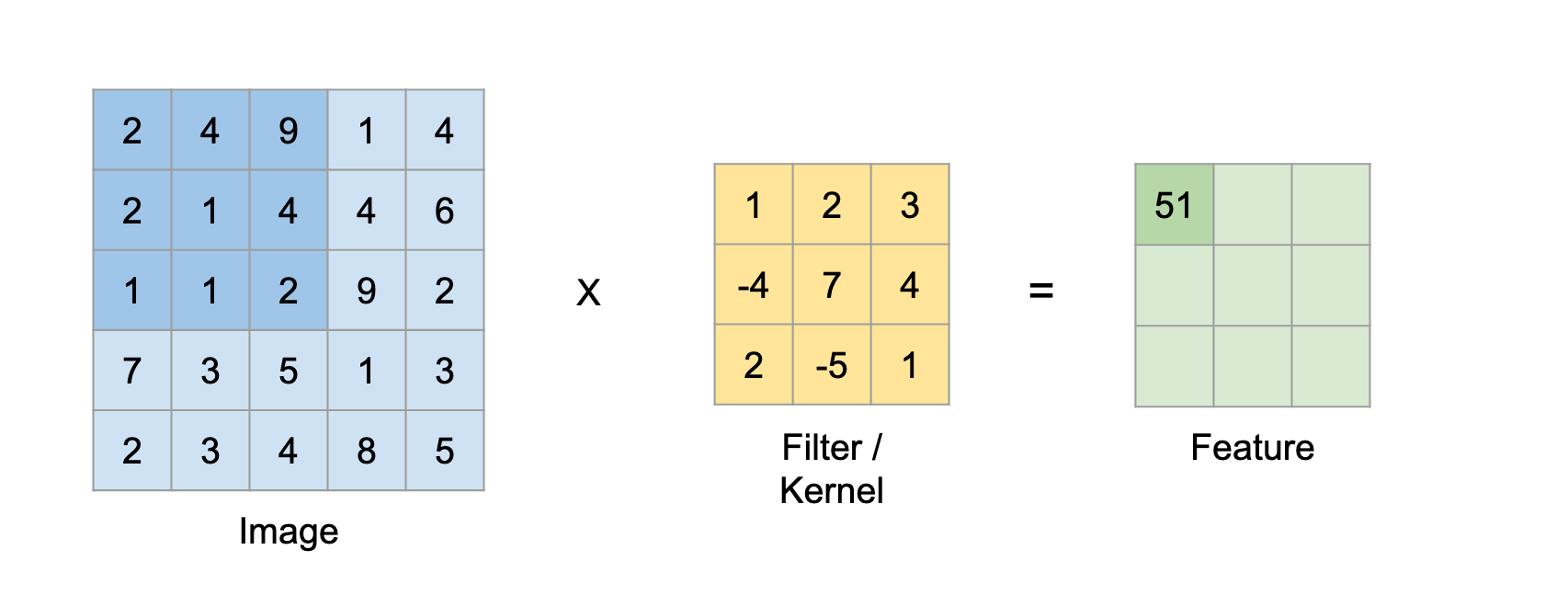


Let’s go layer-wise to get deep insights about this CNN.

First, there a few things to learn from layer 1 that isstriding and padding, let’s see each of them in brief with examples

This in the input matrix of 5×5 and a filter of matrix 3x3, filter is a set of weights in a matrix applied on an image or a matrix to obtain the required features.

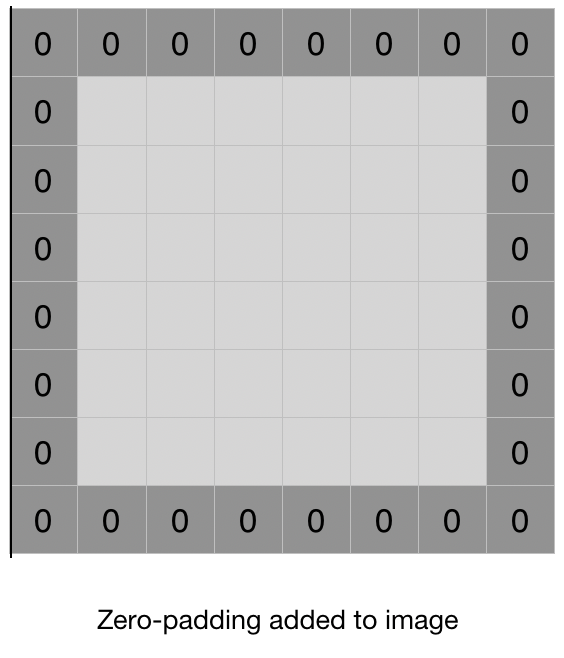
We always take the sum or average of all the values while doing a convolution.



Here the input is of size 5×5 after applying a 3×3 kernel or filters you obtain a 3×3 output feature map.

**6.1. Padding**

While applying convolutions we will not obtain the output dimensions the same as input we will lose data over borders so we append a border of zeros and recalculate the convolution covering all the input values.



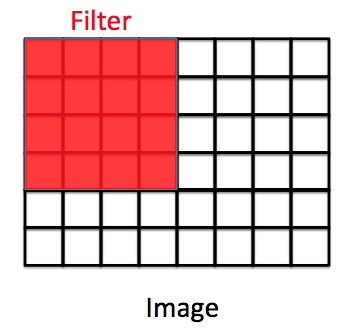
**6.2. Striding**

Sometimes we do not want to capture all the data or information available so we skip some neighbouring cells. Here the input matrix or image is of dimensions 8×8 with a filter of 4x4.

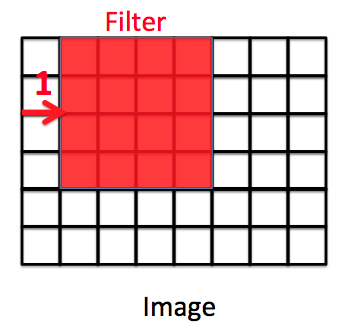
Stride is a component, or tuned for the compression of images and video data. Stride is a parameter of the neural network's filter that modifies the amount of movement over the image or video. For example, if a neural network's stride is set to 1, the filter will move one pixel, or unit, at a time. The size of the filter affects the encoded output volume, so stride is often set to a whole integer, rather than a fraction or decimal.

Basically, a computer read an image from left to right and from top to bottom. Therefore, it starts from the top-left corner all the way to bottom-right corner.

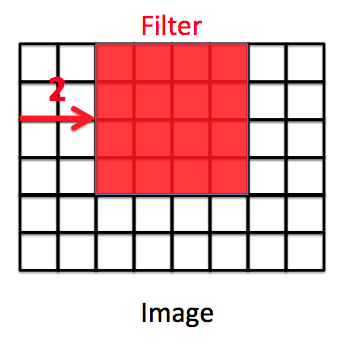
We define how far the filter moves from one position to the next position by “stride”. Let’s look at an example. The red square is a filter. The computer is going to use this filter to scan the image.



If stride = 1, the filter will move one pixel.



If stride = 2, the filter will move two pixels.



**6.3. imageInputLayer:**

An imageInputLayer is the place you initialize the size of input image, here, 100-by-100-by-1 is used. These numbers represent height, width, and the number of channels. In this case, input data is a grayscale image.

**6.4. Convolutional Layer:**

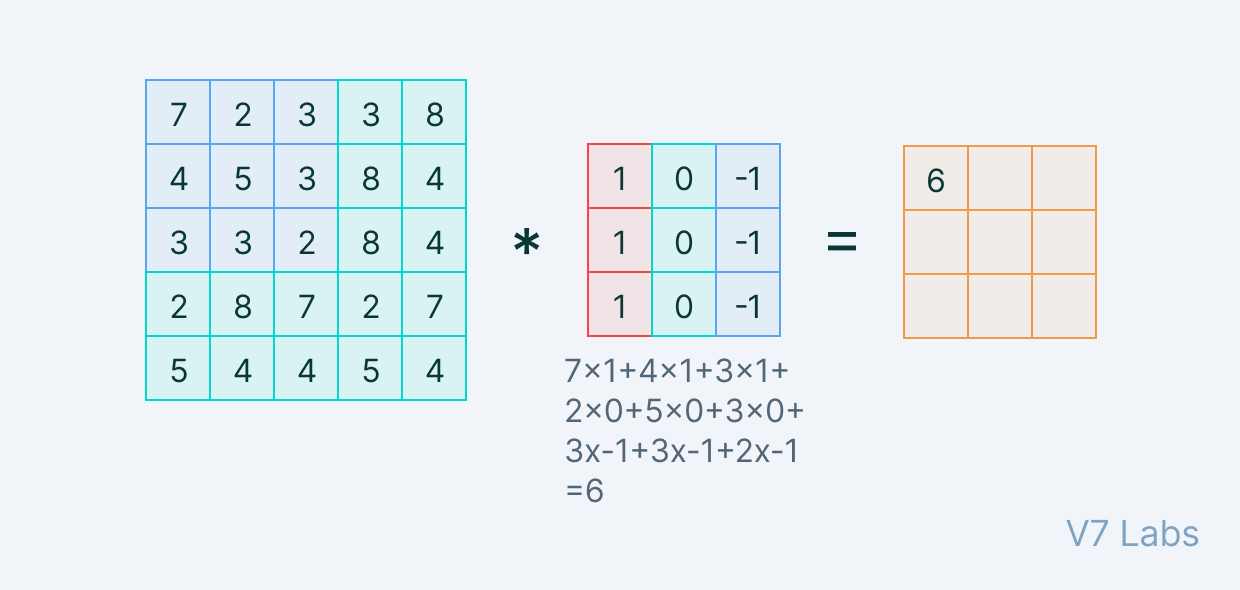
In a convolutional neural network, a convolutional layer is responsible for the systematic application of one or more filters to an input.

The multiplication of the filter to the input image results in a single output. The input is typically three-dimensional images (e.g., rows, columns and channels), and in turn, the filters with fewer rows and columns than the input image. As such, the filter is repeatedly applied to each part of the input image, resulting in a two-dimensional output map of activations, called a feature map.

Keras provides an implementation of the convolutional layer called a Conv2D.

It requires that you specify the expected shape of the input images in terms of rows (height), columns (width), and channels (depth) or [rows, columns, channels]*.*

The filter contains the weights that must be learned during the training of the layer. The filter weights represent the structure or feature that the filter will detect and the strength of the activation indicates the degree to which the feature was detected.

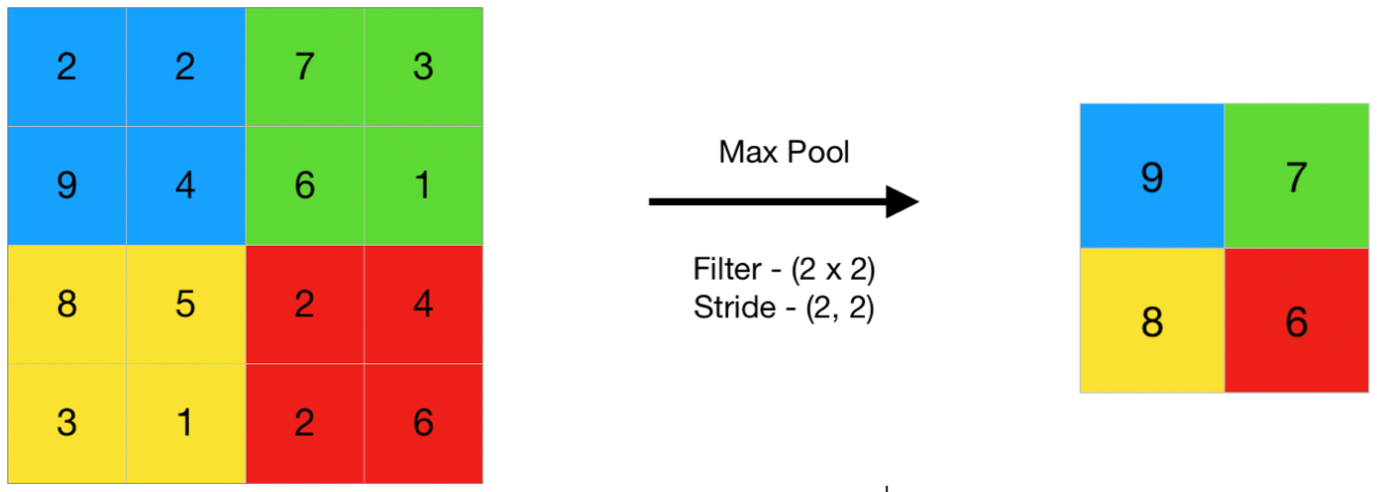


In the convolution layer, we move the filter/kernel to every possible position on the input matrix. Element-wise multiplication between the filter-sized patch of the input image and filter is done, which is then summed.

**6.5. Max Pooling Layer:**

Max pooling layer is one of the down sampling techniques which is used for convolutional layers. In this architecture, pool Size is set to 2 and the number of layers used is 2.

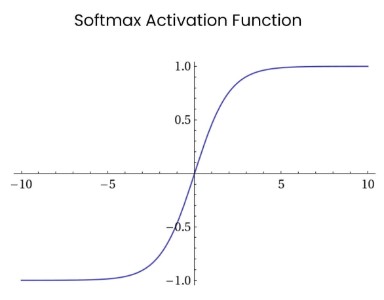
Pooling refers to a small portion, so here we take a small portion of the input and try to take the maximum value termed as max pooling, so by doing pooling on an image we are not taking out all the values we are taking a summarized value over all the values present.



Here this is an example of max pooling so here taking a stride of two we are taking the maximum value present in the matrix.

* 1. **Activation function:**

The activation functions we used here are relu and softmax function. The activation function is a node that is put at the end of or in between Neural Networks.They help to decide if the neuron would fire or not.

****

* + 1. **Rectified Linear Unit (ReLU):**

This function simply returns 0 if your value is negative else it returns the same value you gave, nothing but eliminates negative outputs and maintains values between 0 to +infinity.

The main advantage of using the ReLU function over other activation functions is that it does not activate all the neurons at the same time. This means that the neurons will only be deactivated if the output of the linear transformation is less than 0.

Now how does ReLU transform its input? It uses this simple formula:

f(x)=max(0,x)

ReLU function is its derivative both are monotonic. The function returns 0 if it receives any negative input, but for any positive value x, it returns that value back. Thus, it gives an output that has a range from 0 to infinity.

Let us define a ReLU function

def ReLU(x):

if x>0:

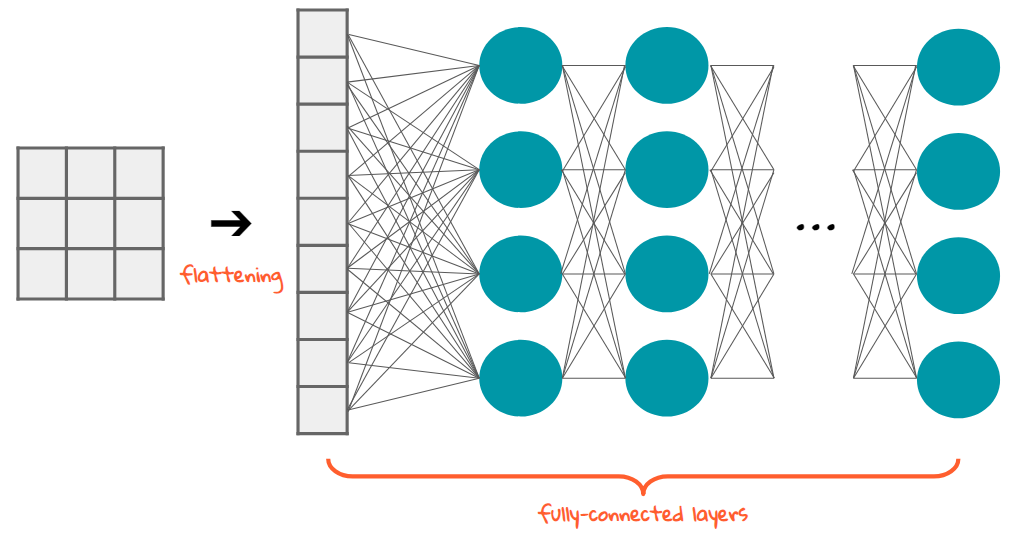
return x

else:

return 0

ReLU is used as a default activation function and nowadays and it is the most commonly used activation function in neural networks, especially in CNNs.  It consists of no heavy computation as there is no complicated math. The model can, therefore, take less time to train or run. One more important property that we consider the advantage of using ReLU activation function is sparsity. Usually, a matrix in which most entries are 0 is called a sparse matrix and similarly, we desire a property like this in our neural networks where some of the weights are zero. Sparsity results in concise models that often have better predictive power and less overfitting/noise. In a sparse network, it’s more likely that neurons are actually processing meaningful aspects of the problem

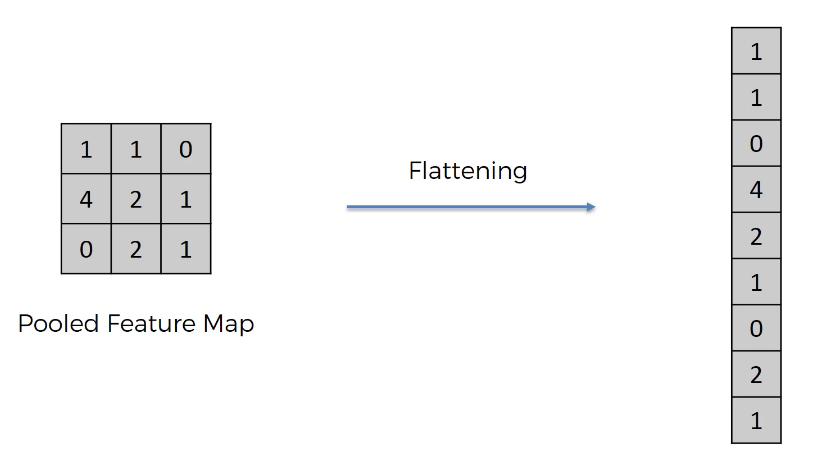
* 1. **Flattening Layer**

Flattening and fully-connected layers are what we have at the last stage of CNN, which means you’re almost there. What are we doing? Image processing. For what? Classifying ‘the gestures.’ We are making a classification model, which means these processed data should be good input to the model. It needs to be in the form of a 1-dimensional linear vector. Rectangular or cubic shapes can’t be direct inputs. And this is why we need flattening and fully connected layer.

Flattening is converting the data into 1-dimensional array for inputting it to the next layer. We flatten the layer of convolution layer to create a single lang feature network. And it is connected to final classification model, which is called fully connected layer.

So, we’ve got the pooled layer, pooled feature map. After we apply the convolution operation to our image and then we apply pooling to the results of the convolution which is the convolved image.

The flattening step is a refreshingly simple step involved in building a convolutional neural network. It involves taking the pooled feature map that is generated in the pooling step and transforming it into a one-dimensional vector. Here is a visual representation of what this process looks like:



The purpose is that we want to later input this into a neural network for further processing.

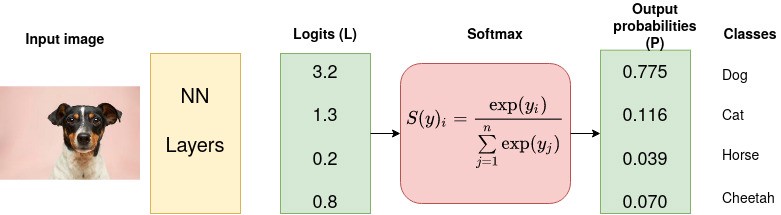
* 1. **Fully Connected Layer:**

Fully connected layers follow max pooling layer. In this layer, all the neurons of all layers are interconnected to the previous layer. The given input argument for this layer is 4, which indicate 4 classes. Fully Connected Layer is simply, feed forward neural network. Fully Connected Layers form the last few layers in the network. The **input** to the fully connected layer is the output from the final Pooling Convolutional Layer, which is flattenedand then fed into the fully connected layer.

* 1. **SoftMax Activation Function:**

It is often used as the last activation function of a neural network to normalize the output of a network to a probability distribution over predicted output class. This function generates an output that ranges between values 0 and 1 and with the sum of the probabilities being equal to 1.

A CNN model which aims at classifying an image as either a Zoom-in, Zoom-out, Rotation, Information (4 possible outcomes/classes). The last (fully-connected) layer of the CNN outputs a vector of logits, L, that is passed through a Softmax layer that transforms the logits into probabilities, P. These probabilities are the model predictions for each of the 4 classes.



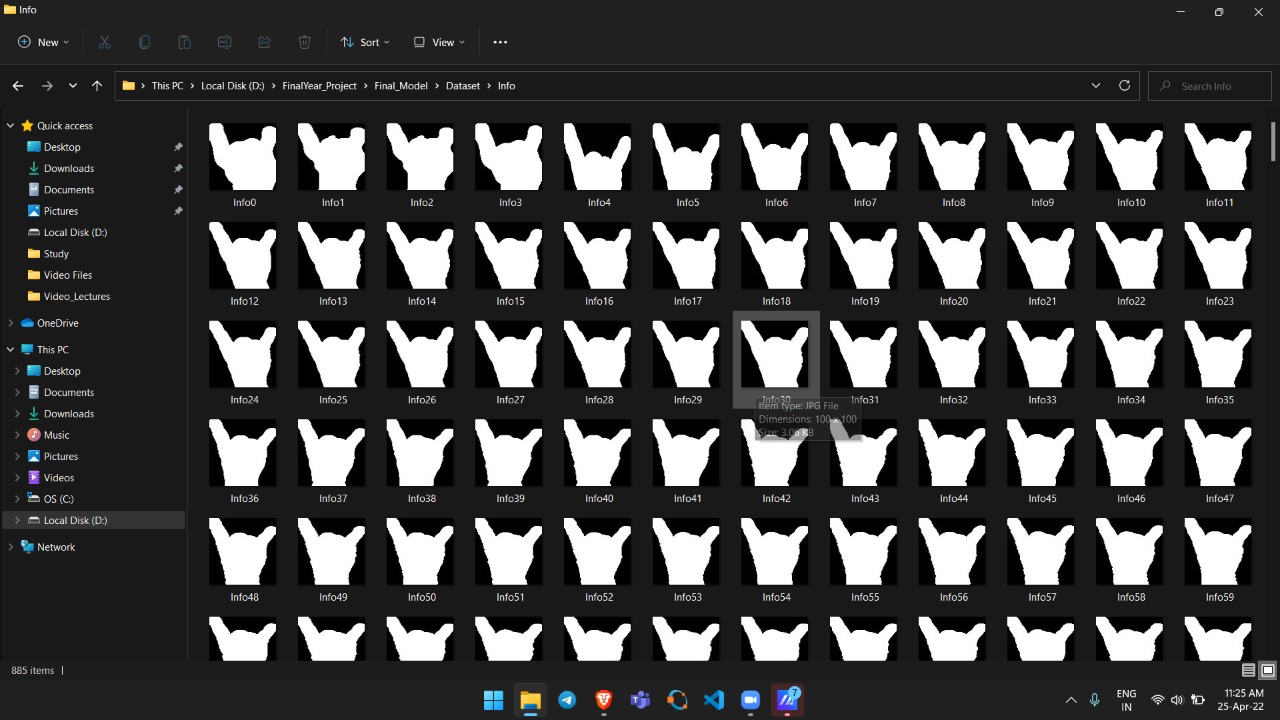
**Zoom-in**

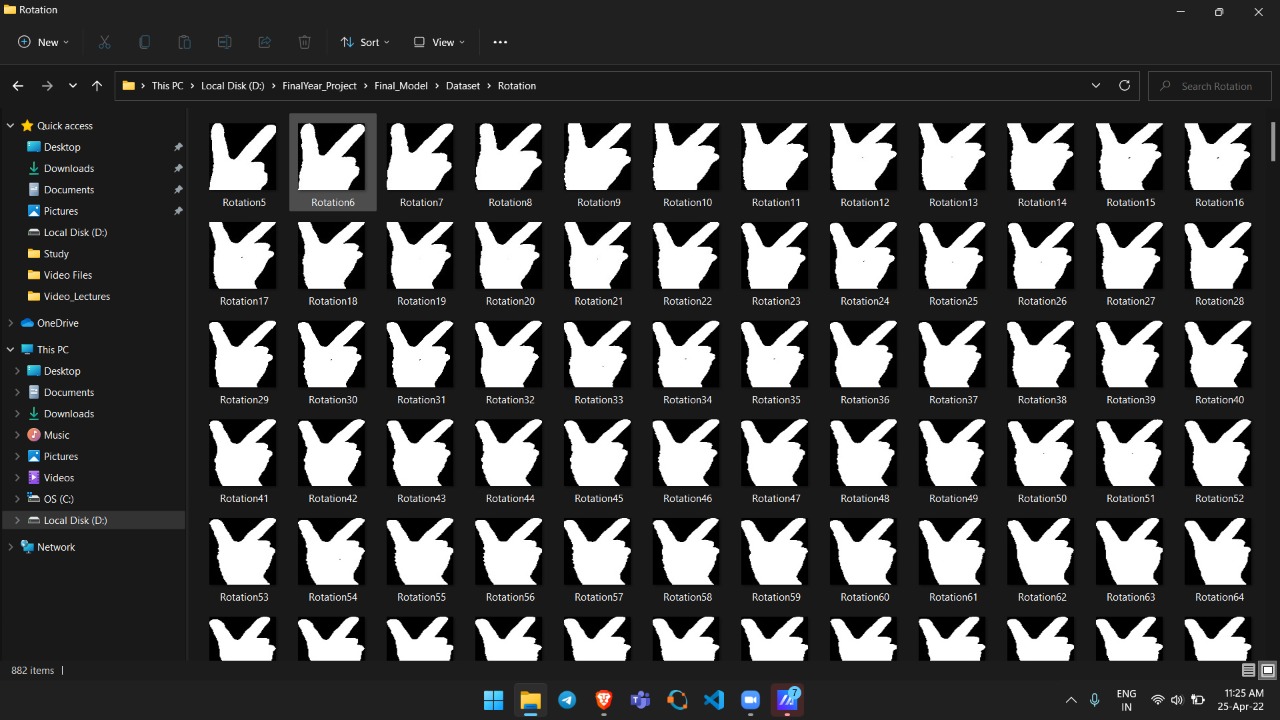
**Zoom-out**

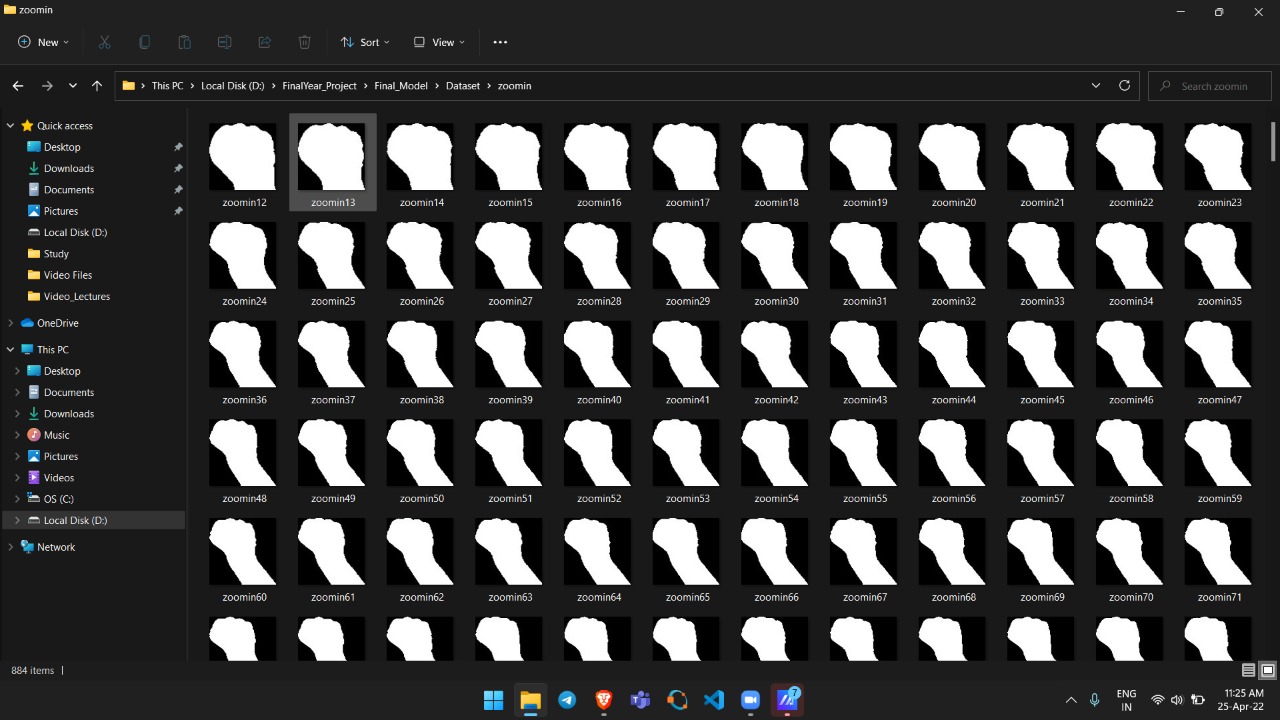
**Rotate**

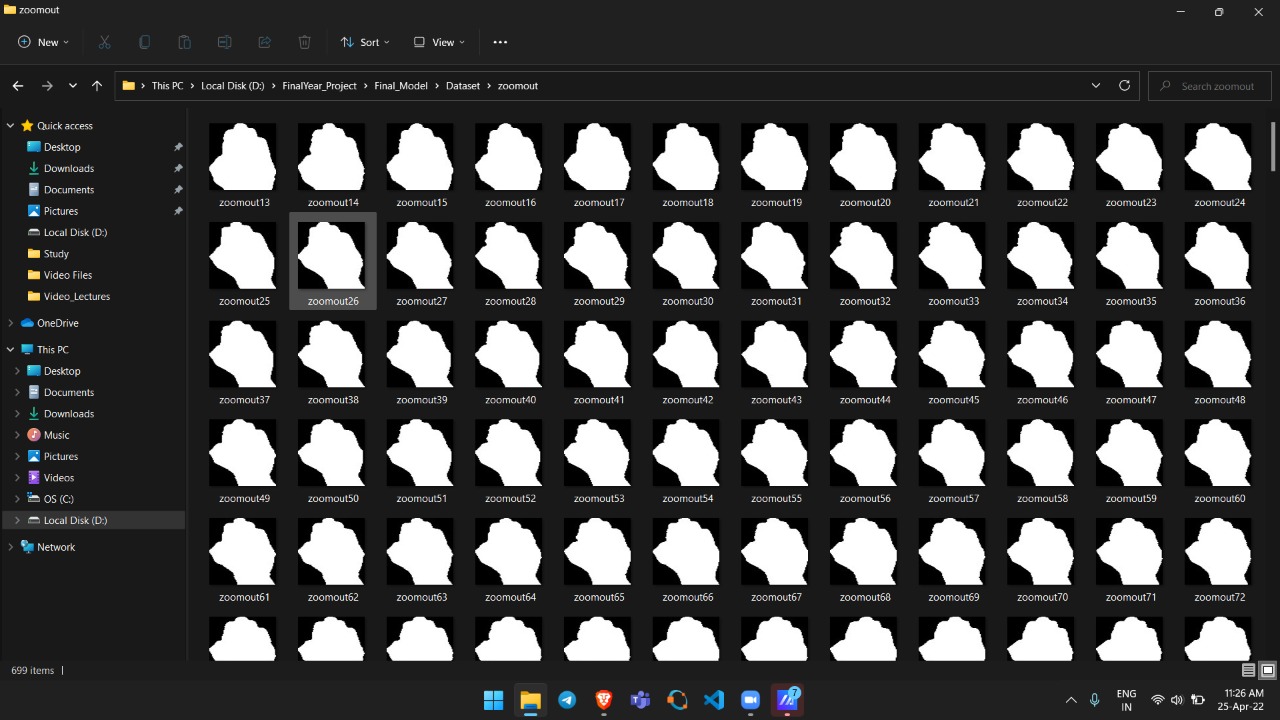
**Information**

1. **Dataset given as input:**

****

****

****



**8.** Below are the snapshots of the Python code to build a CNN architecture using keras library with TensorFlow framework

1. **Import various packages and libraries**

import os  
import numpy as np  
import scipy  
import sklearn  
import keras  
from keras.models import Sequential

from keras.layers import Conv2D

from keras.layers import MaxPooling2D

from keras.layers import Flatten

from keras.layers import Dense

from tool.gesture\_recognition import MotionDetector

import cv2

from keras.preprocessing import image

from keras.preprocessing.image import ImageDataGenerator

from keras.models import model\_from\_json

import h5py

import matplotlib.pyplot as plt

from imutils import paths

import imutils

from PIL import Image

import keyboard

1. **Defining the File Path and loading the image**

Dataset is a dataset of gesture images, with 100X100 grayscale images of 8,000 gestures from 4 categories, and 2,000 images per category. The training set has 6,000 images, and the test set has 2,000 images.

# Load the image, convert it to grayscale, and blur it slightly

for imagePath in paths.list\_images(args["image"]):

image = cv2.imread(imagePath)

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

blurred = cv2.GaussianBlur(gray, (7, 7), 0)

#cv2.imshow("Image", image)

#Load the image

img=load\_img(‘image.jpeg’)

#Report the details of the image

print(type(image))

print(img.format)

print(img.mode)

print(img.size)

#Show the image

img.show()

1. **Resizing all the images**

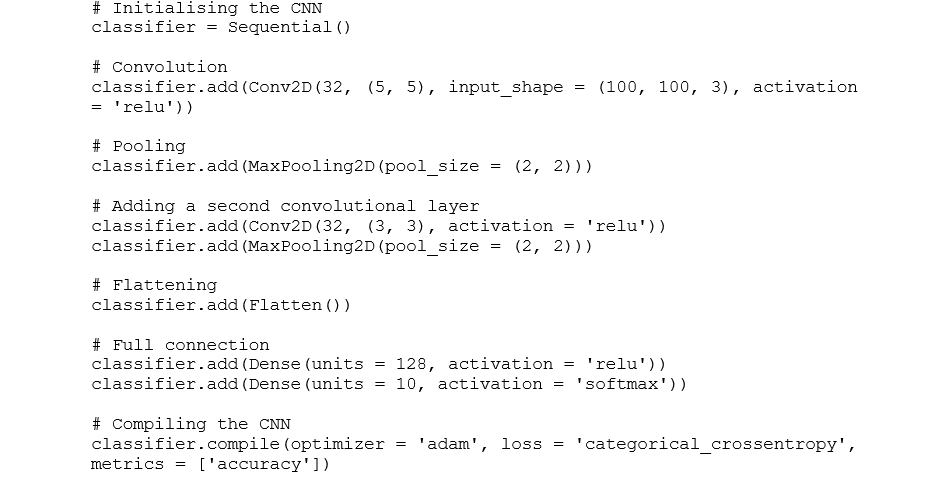
Since neural networks receive inputs of the same size, all images need to be resized to a fixed size before inputting them to the CNN. The larger the fixed size, the less shrinking required. Less shrinking means less deformation of features and patterns inside the image.

for i in range(0,23000):crop\_resized = cv2.resize(image, (100, 100))

1. **Converting images to arrays**

NumPy uses the as array() class to convert PIL images to NumPy arrays . The np.array function also produce the same result. The type function displays the class of images.

images=np.array(images)  
label=np.array(label)

1. **Defining the model**

A convoluted image can be too large and so it is reduced without losing features or patterns, so pooling is done.

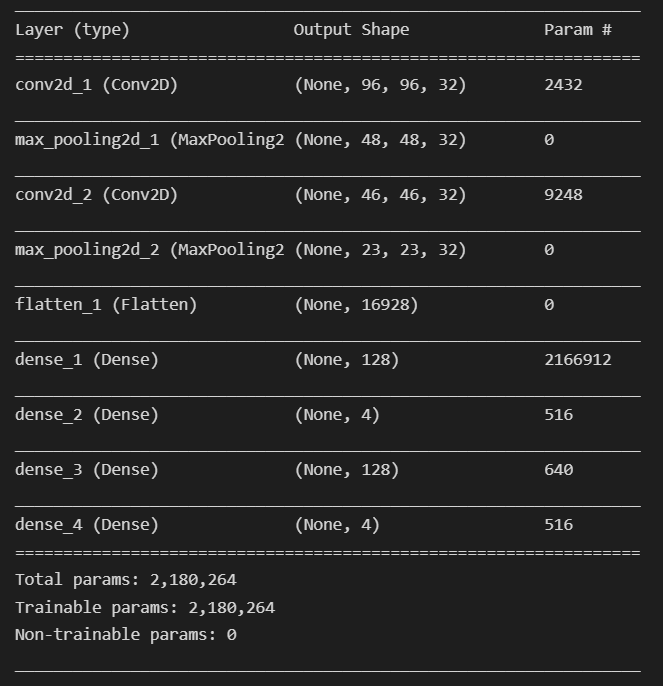
Here Creating a Neural network is to initialize the network using the Sequential model from Keras.

Flatten()- Flattening transforms a two-dimensional matrix of features into a vector of features.

1. **Summary of the CNN model**

model.summary()

It will print the following output:



1. **So now we are required to specify optimizers:**

From tensorflow.keras.optimizers import Adam

classifier.compile(optimizer = 'adam', loss 'categorical\_crossentropy', metrics = ['accuracy'])

Optimizer is used to reduce the cost calculated by cross-entropy. While training the deep learning model, we need to modify each epoch’s weights and minimize the loss function. An optimizer is a function or an algorithm that modifies the attributes of the neural network, such as weights and learning rate. Thus, it helps in reducing the overall loss and improve the accuracy. Adam is replacement optimization algorithm for stochastic gradient decent for training deep learning models. Adam combines best properties of the AdaGrade and RMSProp algorithms to provide an optimization.

The loss function is used to calculate the error

The metrics term is used to represent the efficiency of the model

1. **In this step, we will see how to set the data directory and generate image data:**

From tensorflow.keras.preprocessing.image ImageDataGenerator

# All the images will be rescaled by 1./255

#Flow\_from\_directory function lets the classifier identify the labels from the name of directories the image lies in

train\_datagen = ImageDataGenerator(rescale = 1./255,

shear\_range = 0.2,

zoom\_range = 0.2,

horizontal\_flip = True)

#test\_datagen = ImageDataGenerator(rescale = 1./255)

training\_set = train\_datagen.flow\_from\_directory('dataset',

target\_size = (100, 100),

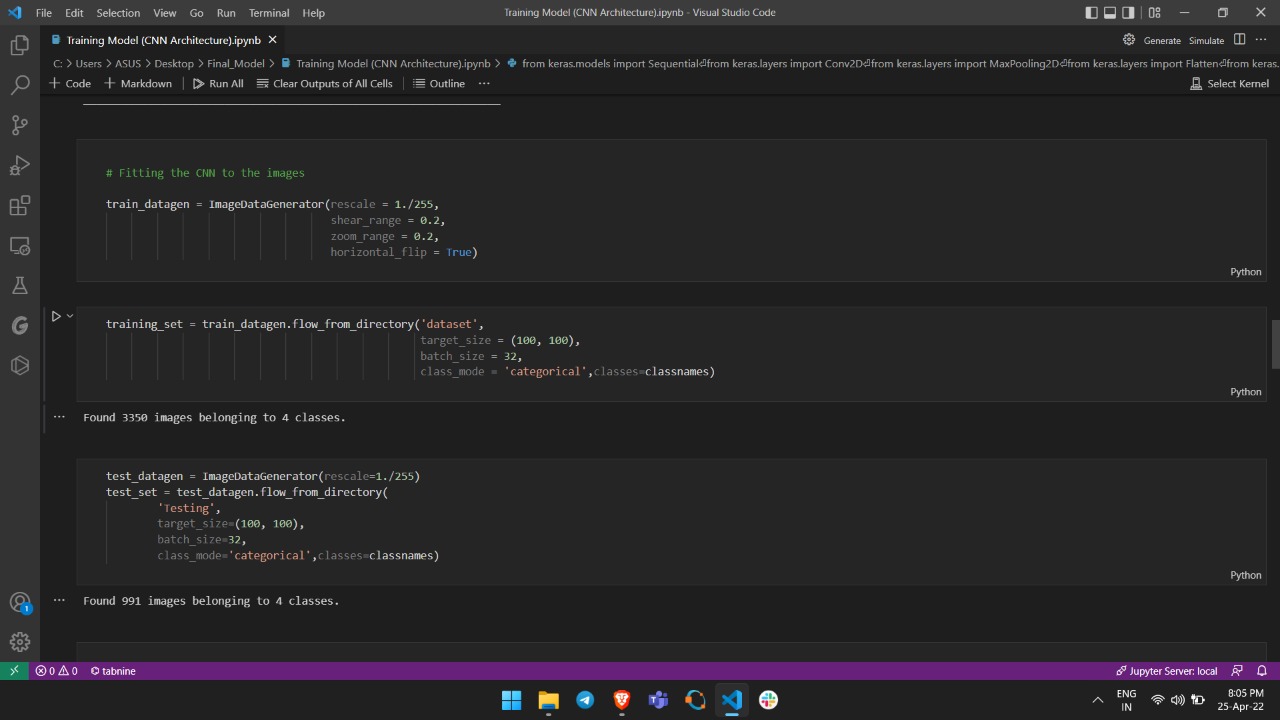
batch\_size = 32,

class\_mode = 'categorical',classes=classnames)

**Batch** – It denotes the number of samples to be taken to for updating the model parameters.

**Sample** – A single row of a dataset.

**Learning rate** – It is a parameter that provides the model a scale of how much model weights should be updated.



**Cost Function/Loss Function** – A cost function is used to calculate the cost that is the difference between the predicted value and the actual value.

**Weights/ Bias** – The learnable parameters in a model that controls the signal between two neurons.

1. **Final step of the fitting model.**

history = classifier.fit\_generator(training\_set,

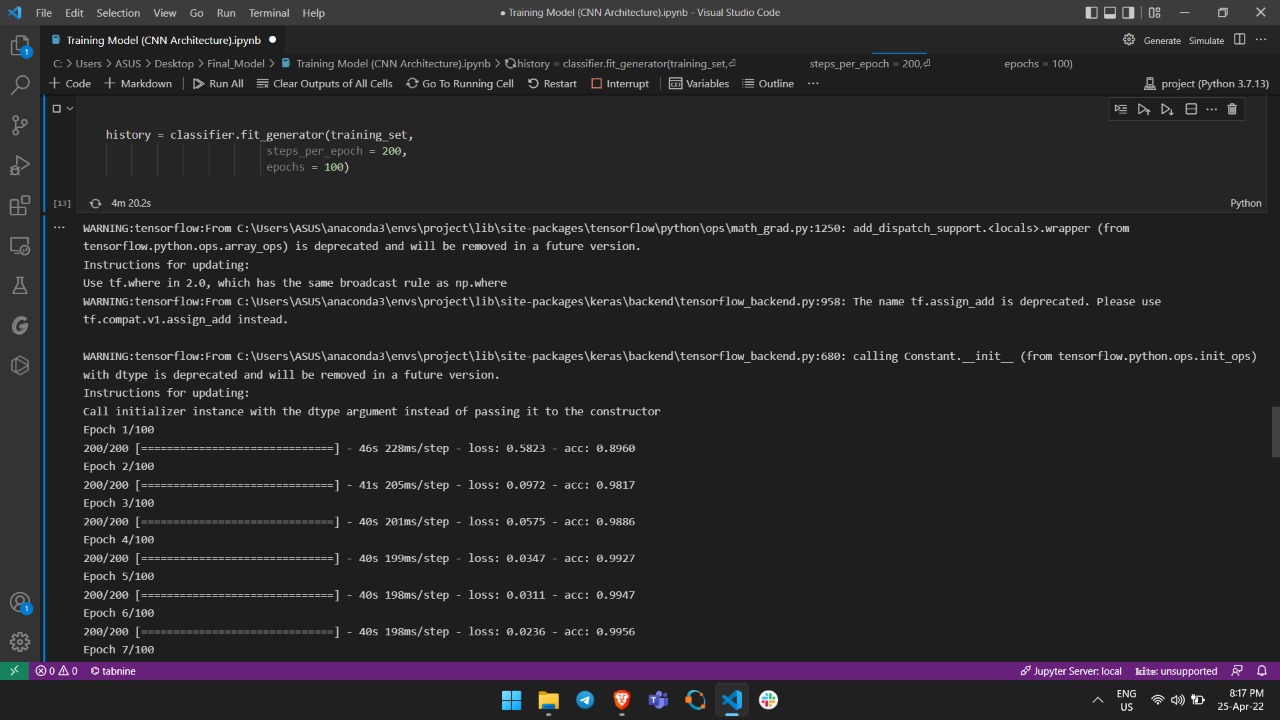
steps\_per\_epoch = 200,

epochs = 100)

**Epoch** – The number of times the algorithm runs on the whole training dataset.

An epoch means training the neural network with all the training data for one cycle. In an epoch, we use all of the data exactly once. A forward pass and the backward pass together are counted as one pass.

An epoch is made up of one or more batches, where we use a part of the dataset to train the neural network.



1. **Python JSON**

JSON stands for JavaScript Object Notation. JSON is a lightweight data format used for data interchange between multiple different languages. It is easy to read for humans and easily parsed by machines.

It’s pretty common for websites to return JSON from API’s so that the information is easy to parse by different programming languages. In Python, the text of JSON is read as quoted-string which contains the value in key-value mapping within { }. Once parsed, it is available as a dictionary object in Python.

Python comes with a built-in package called json for encoding and decoding JSON data.

For working with json type files, first you need to import the json library.

import json

## **Purpose of JSON**

The process of converting python to JSON is done by serialization.

The term ‘serialization’ refers to the transformation of data into a series of bytes (hence serial) to be stored. Since the JSON is read by other languages, various Python-specific objects are converted to a standard JSON acceptable format. For example, a list and a tuple, which are python specific, are converted to arrays when storing it as JSON format.

Likewise, a JSON object is imported and parsed as a python dict.

But, why store it as a ‘dict’ specifically?

Because, the python ‘dict’ is very similar in structure to a JSON, as it can hold other types of data structures within, such as list, tuples, and even other dicts etc

## **How to convert Python Dict to JSON?**

json.dump() and json.dumps() function can be used to accomplish this. The difference between the two is,**‘json.dumps()’ converts a ‘dict’ to a string format whereas ‘json.dump()’ can be used to store it to a JSON file on disk storage.**

In Python Programming, the model type that is most commonly used is the Sequential type. It is the easiest way to build a CNN model in keras Keras Conv2D is a 2D Convolution Layer, this layer creates a convolution kerenel that is wind with layers input which helps produce a tensor od output MaxPooling2D class. Max pooling operation for 2D spatial data. Down samples the input along its spatial dimensions (height and width) by taking the maximum value over an input window (of size defined by pool size) for each channel of the input. The window is shifted by strides along each dimension. Flatten is used to flatten the input. For example, if flatten is applied to layer having input shape as (batch size, 2,2), then the output shape of the layer will be (batch size, 4). The Dense layers are the ones that are mostly used for the output layers. OpenCV is used for all sorts of images and video analysis, like facial recognition, license plate reading, photo editing and lot more. The h5py package is a Pythonic interface to the binary data format. It lets you store huge amounts of numerical data, and easily manipulate that data from NumPy.

It consists of layers. The first layer consists of an input image with dimensions of 100×100. It is convolved with filters of size 5×5 resulting in dimension of 96x96. The second layer is a Pooling operation which filter size 2×2 with the number of filters 32. Hence the resulting image dimension will be 48x48x32.

Similarly, the third layer also involves in a convolution operation with 32 filters of size 3×3 followed by a fourth pooling layer with similar filter size of 2×2. Thus, the resulting image dimension will be reduced to 23x23x32. Followed by a Flatten layer which is usually used as a connection between Convolution and the Dense layers. The Dense layers are the ones that are mostly used for the output layers.

The ReLU activation function used in between these layers.

The final layer will be a softmax output layer with ‘n’ possible classes depending upon the number of classes in the dataset.  ‘Softmax’ which gives a probability for each class and they sum up totally to 1. The model will make its prediction based on the class with highest probability. Here in our model, we have four classes that are Zoom-in, Zoom-out, Rotate and Information.

1. **Future Scope:**

The hand Gesture recognition is moving at tremendous speed for the futuristic products and services and major companies are developing technology based on hand gesture system and that includes companies like Microsoft, Samsung, Sony and is includes the devices like Laptop, Hand held devices, Professional and LED lights. The verticals include where the Gesture technology is and will be evident are Entertainment, Artificial Intelligence, Education and Medical and Automation fields. And with lot of Research and Development in the field of Gesture Recognition Field, the use and adoption will become more cost effective and cheaper. It’s a brilliant feature turning data into features with mix of technology and Human wave. Smart phones have been experiencing enormous amount of Gesture Recognition Technology with look and views and working to manage the Smartphone in reading, viewing and that includes what we call touch less gestures. Google Glass has been embedded into smart televisions nowadays as well, which can easily control and managed by Voice and Hand options. In the medical fields Hand Gestures may also be experienced in terms of Robotic Nurse and medical assistance. As the Technology is always revolving and changing the future is quite unpredictable but we have to be certain the future of Gesture Recognition is here to stay with more and eventful and Life touching experience.

Additional future work will include recognition of dynamic two-handed manipulation gestures for zooming an image, rotating an image etc. We are interested as well as, to experiment with large gesture vocabularies to enhance the interaction flexibility to the system.

The purposed hand gesture recognition system used to recognize the tasks given to computer for processing the image can be extended to recognize gestures facial expressions.

1. **Conclusion:**

Nowadays, application need several kinds of image, pattern as source input of information for analyzing. Serval features of input are to be extracted as to perform various application or tasks. When a transformed from one form to another such as digitizing, scanning, communicating and storing etc. degradation occurs. Therefore, the output has to undertake a process called image enhancement which contains a group of methods that seek to develop the visual presence of an image.

Variable features of gestures are extracted for different input gesture, which indicate the actual nature of the code. The extracted features are enough to represent the given task.

In the project, we consider a vision-based system that can interpret a user’s gesture in real time to manipulate windows & object within a medical data visualization. A hand segmentation procedure first captures the picture of gesture, extracts the feature. Dynamic navigation gestures are translated to commands based on their relative position on the screen. Static gesture poses are identified to execute non-directional commands. The is accomplished by using open palm, feast, victory symbol like features to represent the zoom in, zoom out, rotation, image information. These features are then input to the for classification.

The hand region is detected from the background by the background subtraction method. Then, the palm &finger are segmented. On the basis of segmentation, the fingers in the hand image are discovered and recognized. The recognition of hand gestures is accomplished by a simple CNN classifier. The performance of our method is evaluated on a data set of hand images. The experimental results show that our approach performs well & is fit for the real-time applications.

The performance of the proposal method highly depends on the result of hand detection. If there are moving objects with the colour similar to that of the skin, the object exits in the result of hand detection & then degrades the performance of the hand gesture recognition.

The utilization of gesture recognition technology in the medical environment has been progressing. This project is about the case study on touchless communication between humans & computers to avoid the risk of contamination of germs during surgery.