International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-7 Issue-5S2, January 2019

Emotional Analysis Using Image Processing

U.M Prakash, Pratyush, Pranshu Dixit, Anamay Kumar Ojha

Abstract: ***In machine learning, a convolutional neural network (CNN or ConvNet) is a part of deep and feed-forward artificial neural networks that has successfully visualized images.CNNs use a variation of multilayer perceptron designed to require minimal pre-processing. CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filtering that was hand-engineered in other algorithms. This independence of human efforts for feature design is a major advantage due to which we are using it in our paper. In the context of machine vision, image recognition is the capability of software to identify people, places, objects, actions and writing in images. When we are using our algorithm train the model from our data set of around 600 images, we are getting an accuracy of 85.23%. We can also use other methods for modelling in for this problem set.***

Keywords: Filter, kernel size, convolving, activation map, feature map, stride, max pool, activation function, reception field, epoch cycles.

I. INTRODUCTION

In the 21th century, Stress Management has been a great matter of concern. It's a collection of methods aiming to control a person's state of stress, especially chronic stress, usually for the purpose of improving day to day life. Stress produces various physical and mental disorders which can vary according to individual's situational factors.

These can include health issues as well as depression. One of the important means through which humans display their emotions is through their facial expressions. Though nothing is said verbally, there is more to understand about the messages and signs we send and receive through the use of nonverbal communication like the facial expressions. The process of stress management is considered one of the keys to be happy, satisfied and successful in life.Although life can have various challenges that can be proven difficult to face, stress management provides a number of ways to manage anxiety and maintain overall well-being.

Although humans can recognize facial expressions of their known without any difficulty or delay, reliable expression recognition by machine is still a challenge. There have been several researches in the past few years in terms of feature extraction, face detection and methods used for expression classification, but development of an automated system that accomplishes this task is difficult. In this paper, we used an approach including the Convolutional Neural Networks (CNN) over the face images to study the emotions. The input into our system is an image taken from webcam, and then we use CNN to predict the emotion label which should be one of these labels: anger, fear, happiness, and sadness.

In this, we follow the line of work that applies convolutional neural networks (CNNs) and Image recognition. Our aim is to have an analysis of emotions using images of the person. Convolution neural network model takes an image dataset as input. It is trained by using hundreds of training datasets. The images used for training are basically the Google images which are labelled according to their types. After going through certain layers of CNN for image processing we obtain four different types of emotions that are angry, fear, happy and sad. This model proposes to help stressed individuals by enabling people close to them to recognize when they are in the grip of depression, help them coupe from it, and create natural positive feelings to reduce their stress. Our project can have many modules but for this paper we have finished three modules. Those are Creation of images data set, making a model trained by this data set, and enabling a camera snapped image to be the testing image for the output. Basically, we are looking forward for the creation of a platform which can run over a device making it feasible to function according to the mood of the person by processing its Macro and Micro expressions as mentioned in [7].

II. RELATED WORK

Emotion analysis using deep learning Several methods for person's emotional analysis have been proposed recently. In Emotion Detection and Sentiment Analysis of Images, by Vasavi Gajraj and Aditi gupta at Georgia Institute of Technology worked over the emotion analysis using Deep learning over the images on social media for predicting the mood of person choosing the image. In EEG-Based Emotion Recognition Using Deep Learning Network with Principal Component Based Covariate Shift Adaptation written by Suwicha Jirayucharoensak, Setha PanNgum, and Pasin Israsena and published in 2014 They used Deep learning network to analyse the non-stationary EFG signals. In Deep Learning for Emotion Recognition on Small Datasets using Transfer Learning published in 2015 by image dataset from movies were used and it gave an accuracy of 55.6%. They proposed that SVM and Naïve Bayes can also be used for modelling. In Combining modality specific deep neural networks for emotion recognition in video in 2013, presented for the wild Challenge deep convolutional network was used for analysing the emotional from a video clip in which the combined the emotion analysis from images and audio

Revised Manuscript Received on Janurary 25,2019

Mr. U.M Prakash, Dept. Computer Science, SRM IST Delhi, India
Mr Pratyush, Dept. Computer Science SRM IST Delhi, India
Mr Pranshu Dixit, Dept. Computer Science SRM IST Delhi, India
(**Mr. Anamay Kumar Ojha,** Dept. Computer Science SRM IST Delhi, India

III. IMPLEMENTATION

Image dataset:

The training of convolutional neural network requires thousands of images as their training dataset. Here, we have used 500 to 600 “labelled” images to train our data set. These are the images which we have collected from various sources including Google images and research dataset from other papers. We accumulate these images in a single folder naming it training data set. Then we use python code to shuffle these images and convert it to a Numpy array. So, we can use this Numpy array as our input for the Convolutional neural network.

Fig. 1 Image dataset

Machine learning:

Convolutional Neural network: The convolutional neural network requires the following stages to implement the image processing:-

1. Convolution
2. Pooling
3. Fully connected layer

1. **Convolution layer:** This is the first layer of Convolutional neural network. In this case we are having our input as a70X70 matrix of pixel values. In the convolutional layer basically, the input matrix is read with a **filter.** This is it is choosing a sub matrix from this matrix and then this filter is also a matrix of numbers (the numbers are called **weights** or **parameters**). The area to be selected by the filter is a square matrix whose area is decided by the **Kernel size** that is given. The area selected is known as **reception field.** The Kernel size used is 2. So, a 2X2 reception field will formed. Then, the reception field is changed as the filter starts shifting. This sliding is known as **convolving.**As the filter is sliding, or **convolving**, around the input image, it is multiplying the values in the filter with the original pixel values of the image. These multiplications are all summed up and hence give us a single value. This we are having 32 filters. After sliding the filter to all locations a matrix is formed which is known as **activation map** or **feature map**. For, this problem in python **ReLu (Rectified Linear unit) activation function** is used for the implementation.
2. **Pooling layer:** This is the layer having the input as the output of the convolution layer. To understand pooling the below shown image is the example in which the **stride is 2** so the filter which is 4X4 is divided into 2X2 and out of each 2X2 the maximum element is selected. This is known as **Max pool.** Its function is to reduce the amount of parameters and computation in the network, and hence to control over fitting.

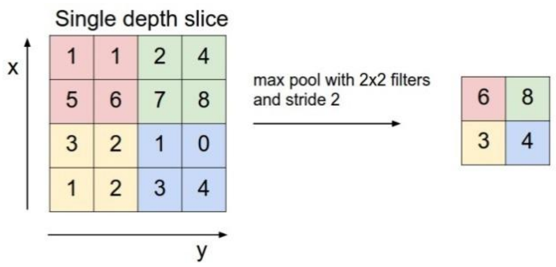


Fig. 2 Pooling Matrix

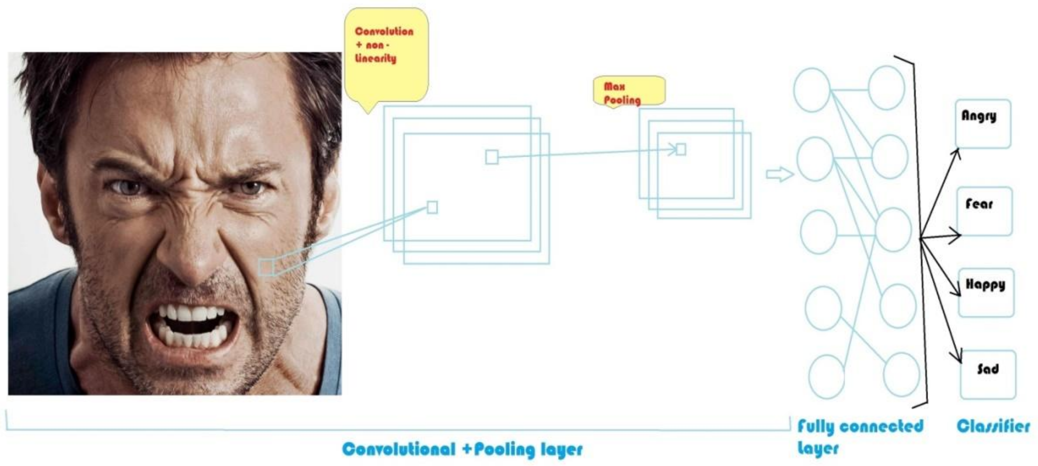


Fig. 3 Working of CNN(Convolutional neural network)

3. **Fully-connected layer:** This is the layer having full connections at all the activations of the previous layer. Hence, the activation can be computed with the bias offset added as it is in the normal neural network.This is the formula for **activation function** for the neural network. By following this **fully-connected layer** is functioning. In this layer the **epochcycles** are updating the activation function according to the variation in the **expected result** and **calculated result.**

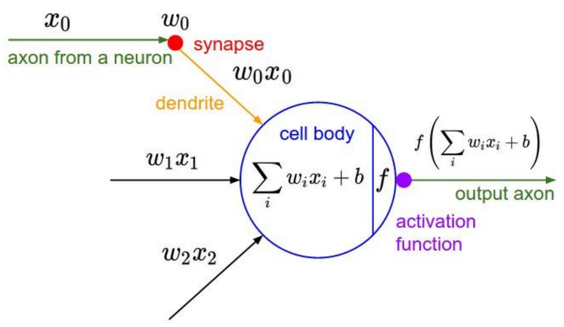


Fig. 4 Activation Function

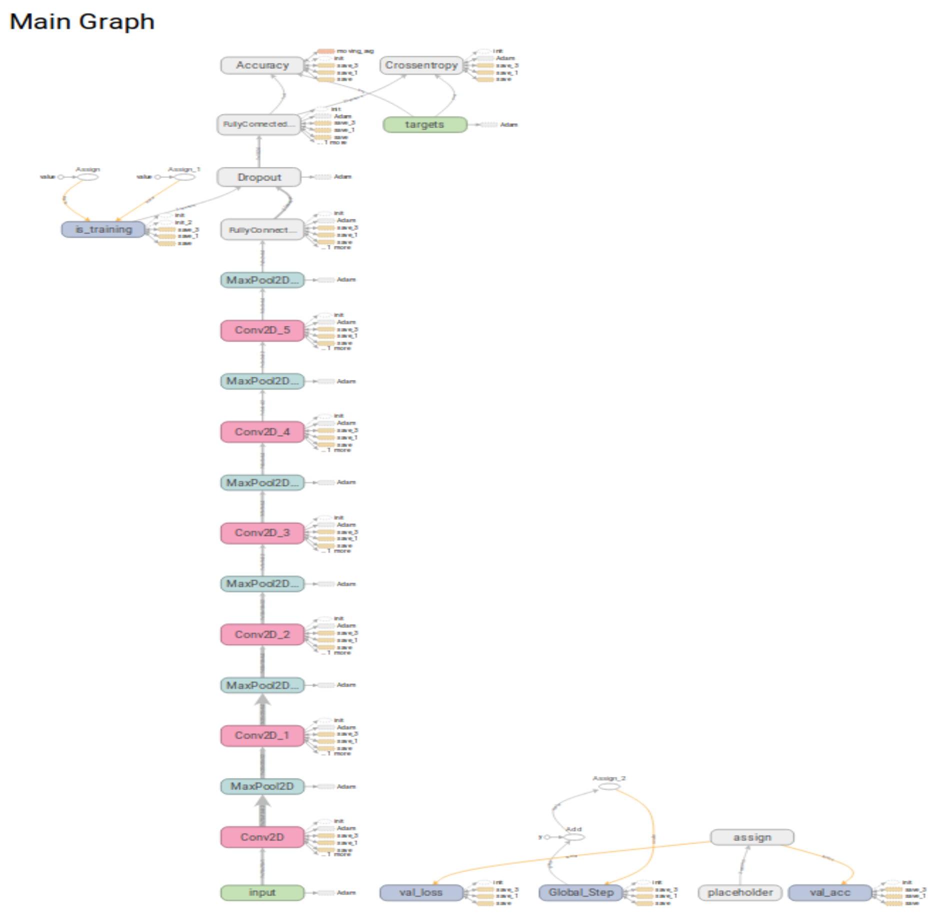


Fig. 5 (Graph on Tensor board)

Model Details

The model constructed for functioning the whole process is basically, a set of **6 convolution+pooling layers** which is done to get a highly detailed **feature map** for the more accurate results as for face image processing a highly detailed feature map is required.

Then it is followed by a **fully connected layer** to process the feature map and give the output as the most probable emotion out of the four (Angry, Fear, Happy, Sad).

IV. EXPERIMENTAL

Training a convolutional neural network

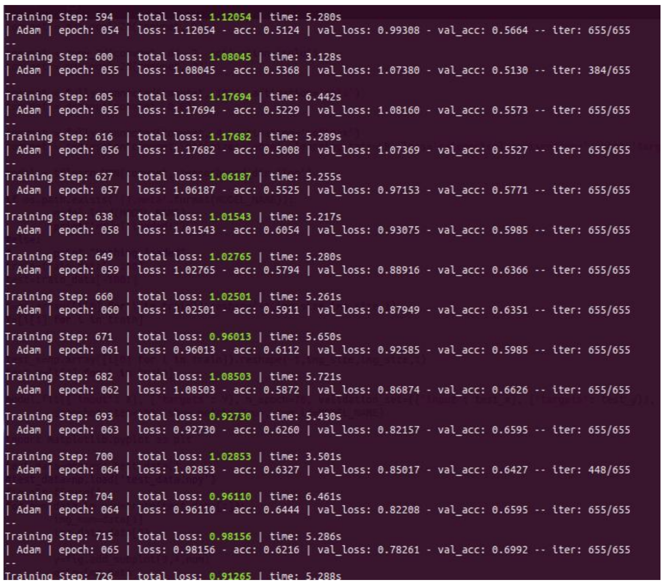
The basic working of a convolutional neural network involves scanning the input image through a **filter** (also called a **neuron** or **kernel**).this filter forms the first convolutional layer and works as feature maps, to scan the input for specific features (such as curves, lines etc.). Now, a basic question arises, i. e how do the kernels in the first **convolutional layer** know how to look for any specific features. Also, the process by which values are allotted to the filters also require explanation. All these tasks are accomplished through a process called **back propagation**. Before any input image is given to the cnn, i.e when it is fresh, the filter values are assigned randomly.to train a cnn, images with **"labels"** are used. These labels act as identifiers, i.e. Upon analysing the labelled image, the cnn **"learns"** how to classify images. In our project, we label the images as happy, sad, angry, fear etc. Back propagation operates in 4 sub processes, which are named as:--

1.) Forward pass: In this process, we pass the training image as an input to the 1st convolutional layer (in our case, a jpeg or png image).generally, an image of size **32x32x3** is passed. These images are labelled, and upon being passed through the network, a haphazard output is generated, such as

[1.1.1.1.1.1.1.1.1.1.1.1] or [0.0.0.0.0.0.0.0.0.0.0.0]

Such an input does not give priority to any number as such. In this state, the network is not able to classify the image and predict the outcome; the above output is then passed to the next **loss function.**

2.) Backward pass: In this phase, we backtrack through the network, in order to find out which weights contribute the most to the loss incurred in the loss function generated earlier. This is done through derivatives, which adjust the weights accordingly to reduce the loss. Once done, we proceed to the last stage



```
Training Step: 594  | total loss: 1.12054 | time: 5.280s
| Adam | epoch: 054 | loss: 1.12054 - acc: 0.5124 | val_loss: 0.99308 - val_acc: 0.5664 -- iter: 655/655
--
Training Step: 600  | total loss: 1.08045 | time: 3.128s
| Adam | epoch: 055 | loss: 1.08045 - acc: 0.5368 | val_loss: 1.07380 - val_acc: 0.5130 -- iter: 384/655
--
Training Step: 605  | total loss: 1.17694 | time: 6.442s
| Adam | epoch: 055 | loss: 1.17694 - acc: 0.5229 | val_loss: 1.08160 - val_acc: 0.5573 -- iter: 655/655
--
Training Step: 616  | total loss: 1.17682 | time: 5.289s
| Adam | epoch: 056 | loss: 1.17682 - acc: 0.5008 | val_loss: 1.07369 - val_acc: 0.5527 -- iter: 655/655
--
Training Step: 627  | total loss: 1.06187 | time: 5.255s
| Adam | epoch: 057 | loss: 1.06187 - acc: 0.5525 | val_loss: 0.97153 - val_acc: 0.5771 -- iter: 655/655
--
Training Step: 638  | total loss: 1.01543 | time: 5.217s
| Adam | epoch: 058 | loss: 1.01543 - acc: 0.6054 | val_loss: 0.93075 - val_acc: 0.5985 -- iter: 655/655
--
Training Step: 649  | total loss: 1.02765 | time: 5.280s
| Adam | epoch: 059 | loss: 1.02765 - acc: 0.5794 | val_loss: 0.88916 - val_acc: 0.6366 -- iter: 655/655
--
Training Step: 660  | total loss: 1.02501 | time: 5.261s
| Adam | epoch: 060 | loss: 1.02501 - acc: 0.5911 | val_loss: 0.87949 - val_acc: 0.6351 -- iter: 655/655
--
Training Step: 671  | total loss: 0.96013 | time: 5.650s
| Adam | epoch: 061 | loss: 0.96013 - acc: 0.6112 | val_loss: 0.92585 - val_acc: 0.5985 -- iter: 655/655
--
Training Step: 682  | total loss: 1.08503 | time: 5.721s
| Adam | epoch: 062 | loss: 1.08503 - acc: 0.5872 | val_loss: 0.86874 - val_acc: 0.6626 -- iter: 655/655
--
Training Step: 693  | total loss: 0.92730 | time: 5.430s
| Adam | epoch: 063 | loss: 0.92730 - acc: 0.6260 | val_loss: 0.82157 - val_acc: 0.6595 -- iter: 655/655
--
Training Step: 700  | total loss: 1.02853 | time: 3.501s
| Adam | epoch: 064 | loss: 1.02853 - acc: 0.6327 | val_loss: 0.85017 - val_acc: 0.6427 -- iter: 448/655
--
Training Step: 704  | total loss: 0.96110 | time: 6.461s
| Adam | epoch: 064 | loss: 0.96110 - acc: 0.6444 | val_loss: 0.82208 - val_acc: 0.6595 -- iter: 655/655
--
Training Step: 715  | total loss: 0.98156 | time: 5.286s
| Adam | epoch: 065 | loss: 0.98156 - acc: 0.6216 | val_loss: 0.78261 - val_acc: 0.6992 -- iter: 655/655
--
Training Step: 726  | total loss: 0.91265 | time: 5.288s
```

Fig. 6 Statistic during training

Acc-Accuracy (data during training with 655 images)

3.) Loss function: A label is a part of the image that identifies the features that distinguish it. To illustrate, a label could be the following matrix----- [0 0 0 0 0 1 0 0 0 0 0]

The loss function is computed by the formula, as shown below:

$$E_{total} = \sum 1/2(target - output)^2$$

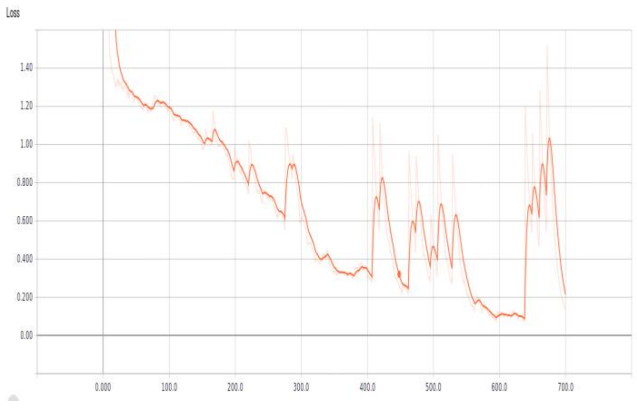


Fig. 7 Loss per epoch cycle (Loss (Y-axis) vs Epoch (X-axis))

Initially, the loss calculated is very high. Naturally we need to minimize it to ensure that correct values are assigned to the layer filters. This is done in a way similar to solving an optimization problem of calculus, as in, we determine the inputs which directly contribute to the loss of the network. This is done in the **backward pass** phase.

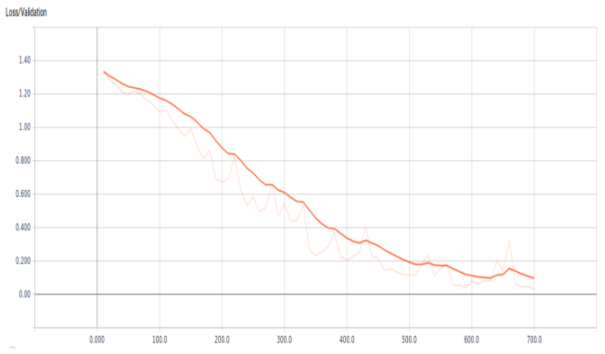


Fig. 8 Total Loss (Loss (Y-axis) vs Epoch (X-axis))

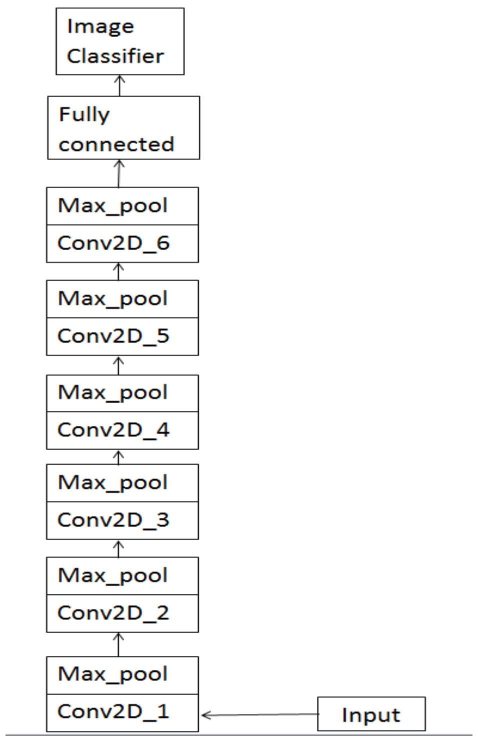


Fig. 9 Flow chart of the model

4.) Weight update: Here, the weights are updated and changed accordingly to suit our purposes. The updates are performed in accordance so that they change in the opposite direction of the **gradient**. This is computed by the learning rate:

$$w = w_i - \eta \frac{dL}{dW}$$

w = Weight
w_i = Initial Weight
η = Learning Rate

Fig. 10 Weight Update

The **learning rate** is chosen by the programmer. The learning rate is set, so as to ensure that it is neither too high, nor too low. A high learning rate ensures faster convergence, but can also lead to jumps that are too large and thus, fail to converge on the right set of values.

The above processes are repeated for a fixed set of iterations, to make the weights converge on a suitable value, upon reaching which the filters are set correctly. Learning rate for this model is **1e-3.**

V. CONCLUSION

In this paper we have majorly covered a method by which we can use the CNN for the image classification of the emotions. Having trained the CNN by a huge image set we can make an ideal model to classify the emotions of a person by his facial image. CNN allows us to extract some complex features from the face and hence is very much helpful for the image classification on the basis of expressions.

VI. SCOPE

Further, the scope of the research done in this paper will be leading to a project in which we will enable a device to work according to the mood of its user. The main purpose for the project is to deal with issues **Stress management, accidental emergencies, etc.**

Stress management: Regarding stress management, we are going to monitor the behaviour of a person over a period of time on what he practises in various kinds of mood. Basically to establish a **Mood-to-action** table and then after the model is trained, the device will be enabled to perform according to the mood of the person. So, whenever a person would be in grief or stress the device would analyse it and then function according to lighten his mood.

Accidental emergencies: We will be working on the algorithm to predict an accident scene the by the same method so that the device can predict the accident scene and can alert the nearby hospital or emergency services by sending them location through GPS.

REFERENCES

1. https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/
2. https://s-media-cache-ak0.pinimg.com/originals/df/47/a7/df47a74b5e1514dcc03a700bd1634a9d.jpg
3. https://c.wallhere.com/photos/91/73/home_alone_macaulay_culkin_kevin_mccallister_boy_fear_shout_fright-795921.jpg!d
4. https://www.askideas.com/media/13/Crying-Baby-Funny-Face.jpg
5. http://img1.showmoreimages.com/pic/21lZGlhLmdldHR5aW1hZ2VzLmNvbS92aWRlb3MvZmFjZS1vZi1hLWhhcHB5LWJveS12aWRlby1pZDEwNDMzODg4Nz9zPTY0MHg2NDAlog
6. https://i.pinimg.com/736x/a1/7b/2d/a17b2dbcaf08929c6c62920db3d44b8e--anger-management-management-tips.jpg
7. http://www.apa.org/science/about/psa/2011/05/facial-expressions.aspx
8. For the video tutorialhttps://youtu.be/27FPv1VHSsQ000000