TIME SEQUENCE BASED GESTURE RECOGNITION

TARA MARY JOSEPH

TABLE OF CONTENTS

I.	SYNOPSIS	3
II.	INTRODUCTION	3
III.	LITERATURE REVIEW	4
IV.	PROCEDURE	8
V.	OBSERVATION	13
VI.	CONCLUSION	13
VII.	FUTURE SCOPE	13
VIII.	REFERENCES	14

I.SYNOPSIS

The problem statement at hand is to recognize different hand gestures that are performed over a real time webcam stream. The gesture images that are captured and stored are used to train and build deep learning models. A real-time stream will be used to assess the model's ability to recognize gestures. The proposed system gives a recognition algorithm to recognize a set of six specific static hand gestures, namely, Pause, Play, Rewind, Restart, FF(Fast Forward), and None.

II.INTRODUCTION

Hand Gesture Recognition has been gaining popularity in improving human computer interaction. It can be used in various applications, including healthcare, music creation, controlling robots, and translating sign language. The most recent computer applications require user participation to be used effectively. In light of this, research into human-computer interaction (HCI) has increased during the past few years. The vast majority of HCI devices currently in use are based on mechanical gadgets like keyboards, mouse, joysticks, or gamepads. To achieve a more natural interaction between computers and humans, certain applications have been developed with the use of hand gestures. In recent years, the utilization of human movements, particularly hand gestures, has grown to be a significant component of human-computer intelligent interaction (HCII), which inspires study into the modeling, analysis, and detection of hand gestures.

Gestures are helpful in computer interaction as they are the most fundamental and expressive means of human communication. Each gesture-recognition algorithm is influenced by the environment, application domain, and user cultural background.

Depending on the input data type, the approach for interpreting a gesture could be done differently. However, the majority of the methods rely on crucial pointers that are visualized in a 3D coordinate system. Depending on the quality of the input and the algorithm's methodology, the gesture can be identified with high accuracy based on the relative motion of these. The ability to recognize gestures as input makes computers more accessible for those with physical disabilities and improves engagement in virtual reality or gaming environments.

Applications of gesture recognition have become more popular over the years. Some fields where this application is more prominent are:

1. Health care: Gestures can be used to manage resource allocation in healthcare facilities, communicate with medical equipment, operate visualization displays, and support disabled individuals throughout their rehabilitation. Gestures can also be used by

surgeons to facilitate mouse functions on computers including pointer movement and button presses.

2. Gaming: The "fast-response" criterion applies to computer vision-based, hand-gesture-controlled games where the system must react swiftly to user motions. In contrast to applications without a real-time requirement, where recognition performance is given top attention, games require computer-vision algorithms to be reliable and effective.
3. Home Appliances: Electronic home devices can be programmed in a way that they could be controlled by hand gestures without the need for a separate control device. A hand gesture-based remote can replace all other household remote controls and carry out all of their functions.
4. Communication: For virtual reality, commanding mechanisms that may be operated covertly are generally favored, such as voice commands, lip-reading, facial expression interpretation, and hand gesture recognition.As hand gesture recognition is at the core of sign language analysis, a robust hand gesture recognition system will be beneficial for the deaf and dumb population to communicate.

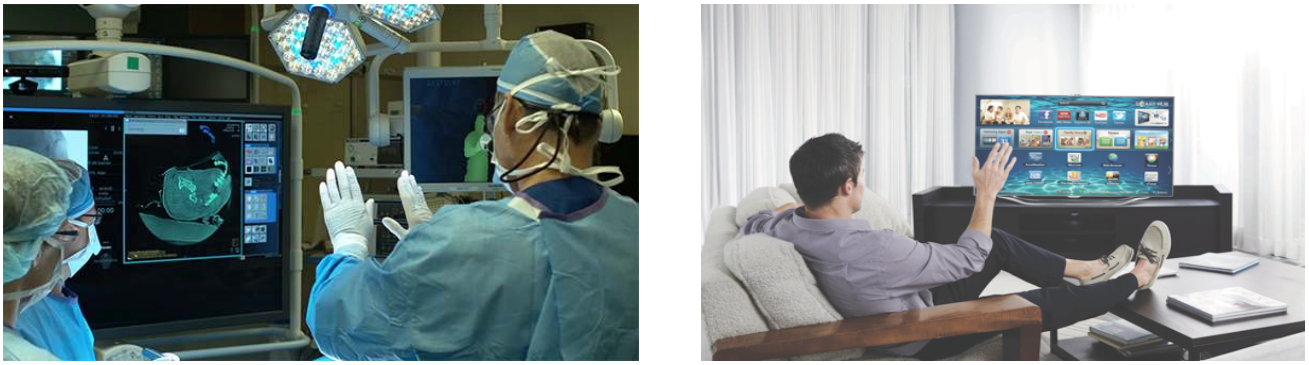


Fig 1. Application of gesture recognition in health care and home appliances

III. LITERATURE REVIEW

Deep learning is a growing field whose applications keep growing over the years. The different types of deep learning models are broadly classified into supervised and unsupervised models. First, most deep learning models have a large model complexity. Image recognition, speech recognition, natural language processing, and many more applications all heavily rely on deep learning models.

Convolutional Neural Networks (CNNs) is an improved and more complex version of artificial neural networks that are designed to handle more complex data pre-processing and computation

tasks.CNN may be the most effective and flexible model for image classification problems because they were created for image data as in our case.

After the input data is imported into the model, there are 4 parts to building the CNN:
1. Convolution: is a method for generating feature maps from our input data. Maps are then filtered using a function.
2. Max-Pooling: this allows our CNN to identify an image even when it is modified.
3. Flattening: Flatten the data into an array so CNN can read it.
4. Full Connection: The hidden layer, which also determines our model's loss function.

CNN eliminates the need for manual feature extraction. This makes these models highly accurate for computer vision purposes.(refer fig.2)

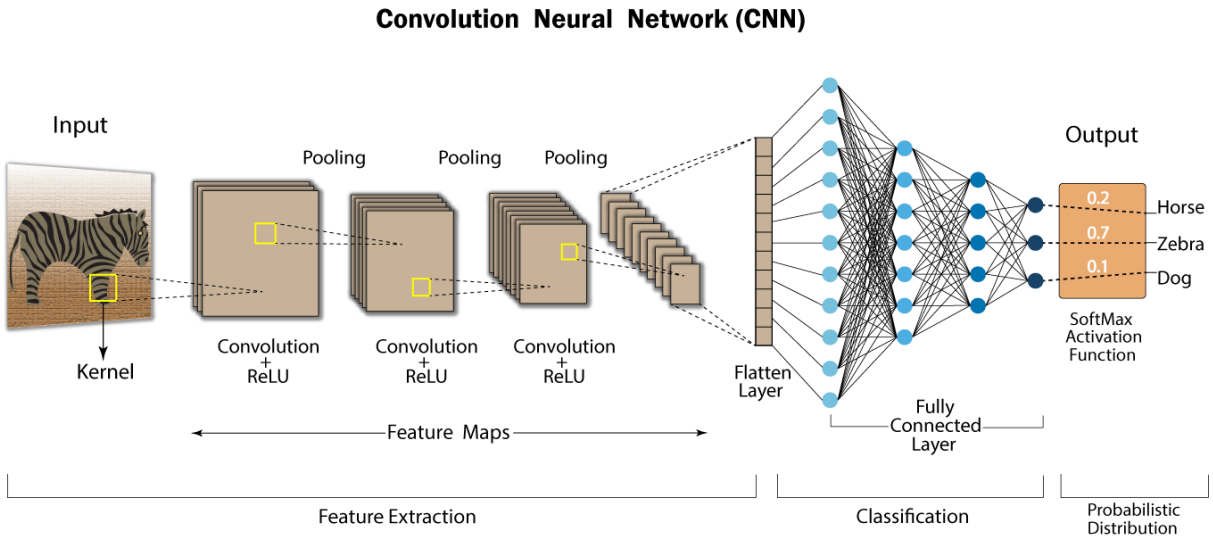


Fig 2. Architecture of a CNN

- ➔ ReLU: The rectified linear activation function, also known as ReLU, is a piecewise linear function that outputs zero if the input is negative and the input directly if the input is positive. It has evolved as the default activation function for many varieties of neural networks since a model that uses it trains more quickly and typically performs better.
- ➔ Softmax: Soft-max is an activation layer that is typically applied to the network's last layer, which serves as a classifier. This layer is in charge of classifying the input into several kinds.

Transfer Learning

Very few individuals actually train a whole convolutional network from the beginning (including random initialization) since it is uncommon to have a dataset large enough. It is more typical to pretrain a ConvNet on a very large dataset (such as ImageNet, which contains 1.2 million images with 1000 categories) and then use the ConvNet as initialization or a fixed feature extractor for the task at hand.

Pretrained models: As modern ConvNets take 2-3 weeks to train across multiple GPUs on ImageNet, people frequently post their final ConvNet checkpoints so that others can use the networks for fine-tuning. The pretrained models used to train our dataset in this project are:

- **ResNet** - is one of the famous deep learning models that was introduced by Shaoqing Ren and his partners. It is a 34-layer plain network in the architecture that is inspired by VGG-19 in which the shortcut connection or the residual blocks are added.

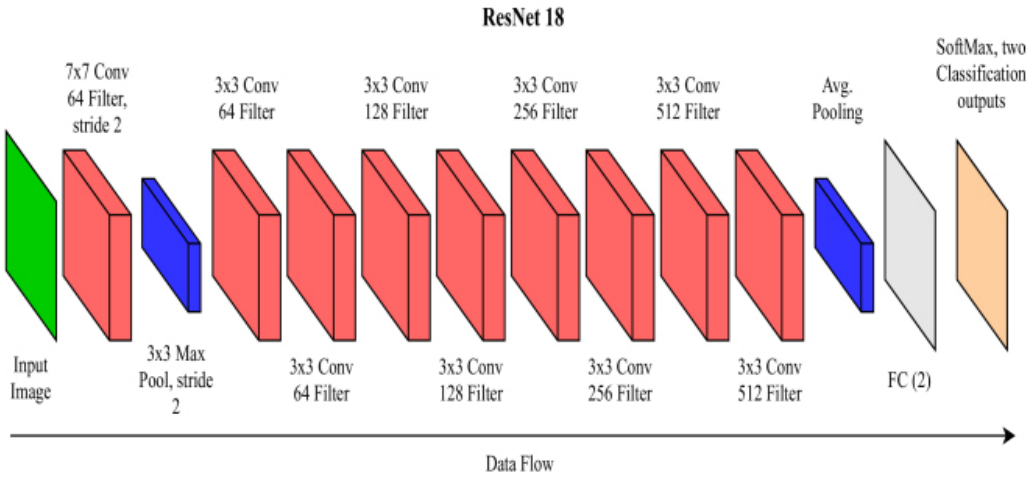


Fig 3.ResNet18

- **AlexNet** - it is an 8-layer CNN that won the ImageNet Large Scale Visual Recognition Challenge 2012 by a large margin with 85% accuracy and was a big breakthrough in the field of computer vision. AlexNet consists of eight layers: five convolutional layers, two fully connected hidden layers, and one fully connected output layer. Also, this model used the ReLU instead of the sigmoid as its activation function.

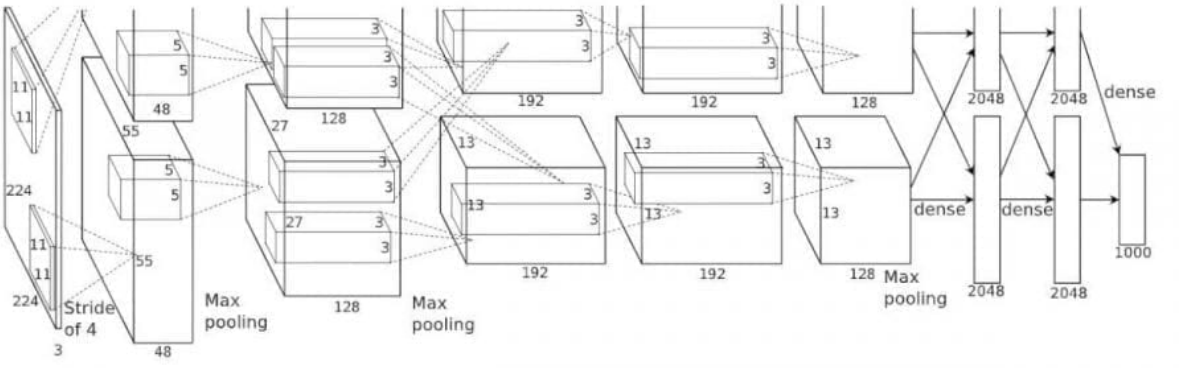


Fig 4.AlexNet

- **VGG16 -** this model made an improvement over AlexNet by using multiple 3×3 kernel-sized filters instead of large kernel-sized filters. The model achieves 92.7% top-5 test accuracy in ImageNet making it one of the best working models for image classification. The 16 in VGG16 refers to 16 layers that have weights. In VGG16 there are thirteen convolutional layers, five Max Pooling layers, and three Dense layers which sum up to 21 layers but it has only sixteen weight layers.

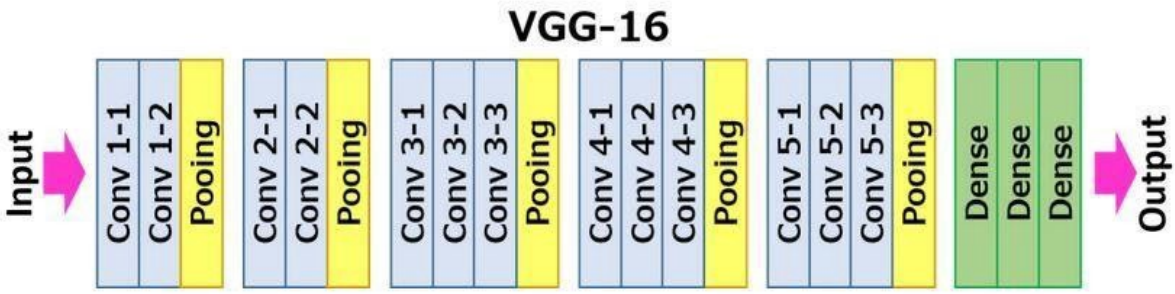


Fig 5.VGG16

Recent methods in vision based gesture recognition systems

Numerous scientific applications, ranging from facial gestures to full-body human movement, have adopted gesture recognition. Hence, various applications have been developed and have created a stronger need for this type of recognition system.

Dong in his paper, Vision-based hand gesture recognition for human–vehicle interaction presented at the Proceedings of the International Conference on Control, Automation and Computer Vision (1998) described a method for vision-based gesture recognition for

human-vehicle interaction in their work. Human tendencies and gesture differentiation were taken into account when creating the models of hand gestures, and human skin tones were employed to segment the hands. The primary emphasis of the research was on the human-vehicle interaction in various circumstances such as summoning the vehicle, stopping the vehicle, directing the vehicle, etc. The only drawback they faced was that in the presence of skin-colored objects in the background, the system's performance was affected.

Hand gesture analysis can be divided into two main approaches, glove-based analysis, and vision-based analysis. The glove-based method uses mechanical or optical sensors that are mounted to gloves to convert electrical data from finger flexing into hand posture. This approach was used by Parvini and Shahabi to recognize ASL signs in their work utilizing mechanical and biomechanical characteristics. The recognition rate was at 75 %.(refer to article [2]).

With each number being represented by a different hand gesture, Swapna proposed another hand gesture recognition system to identify the digits 0 through 10. Image capture, applying a threshold, and number recognition are the three primary phases in this system. It received an 89 percent recognition rate. (refer to article [3]).

In vision-based analysis, it is based on how people perceive their surroundings. In this method, several feature extraction methods are used to extract features from the images. By using nearest neighbor for gesture identification, Freeman and Roth presented their paper Orientation histograms for hand gesture recognition at the International Workshop on Automatic Face and Gesture Recognition (1995). They employed this method to extract the attributes of 10 different hand gestures. The same feature extraction method was applied in another study for the problem of recognizing a subset of American Sign Language (ASL). (refer to article [4]).

IV. PROCEDURE

The overview of the gesture recognition system (shown in Fig. 5) consists of the following stages. The first stage consists of capturing the images by opening the webcam and saving the captured images to the respective folders. The second step makes use of the PyTorch library to make any data modifications and to load the pretrained models through transfer learning. This stage also consists of benchmarking the performance of each of the models and choosing the best model for testing. In the third stage, the gestures are performed again in real time where it is predicted.

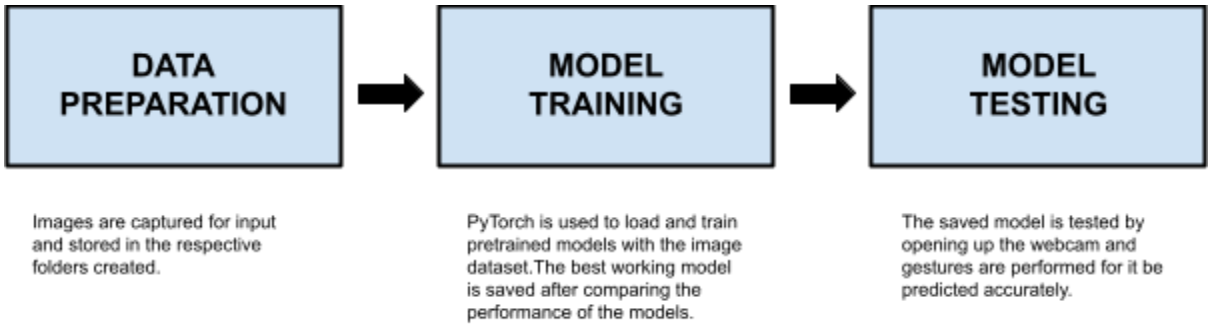


Fig 6. Solution Approach

Stage 1 - Data Preparation

The compilation of a hand gesture database (i.e., the choice of certain hand gestures) typically depends on the application for which it will be used. In our case, the gesture recognition system was developed such that it has future scope in controlling a media player. Hence, the gestures were selected with each one pertaining to a task namely,
Play, Pause, Rewind, Restart, and FF(Fast Forward).

The directory structure was created as follows: Images => Train or validate => gestures
Six folders were created under the train and validate set for five predefined gestures - Play, Pause, Rewind, Restart, and FF along with None folder which consists of a plain background and other gestures(.refer fig 7)

Using the OpenCV package in python, we captured images as frames from a video stream. The laptop's webcam was accessed with the code, where a rectangular frame was created in which the gestures were performed. These gestures were captured and saved with a grayscale filter in the appropriate gesture directories in the train and validate sets.

Play Pause Rewind

Restart FF None

Fig 7. Each gesture images with their respective task and the none folder

A total of 300 images were captured for each gesture and None folder in the train set and 100 images each for the validation set. The images were captured with size 224*224.

Stage 2 - Model Training

In this stage, torchvison package was used to transform the collected images using a few data augmentation techniques such as RandomHorizontalFlip, RandomResizedCrop, etc., and was subjected to normalization and converted to Tensor for the train set. The images in the validate set were subjected to normalization and converted to Tensors as well. A PyTorch tensor is an n-dimensional array (matrix) containing elements of a single data type.

ImageFolder and DataLoader from torchvison were then used to load our image data set and save it to the memory quickly. The images are sent in batches of four for both the train and validate set.

Using torch and torchvison package, the pretrained models were able to be loaded. The optimizer used to optimize the function for each of the models is Stochastic Gradient Descent. In the SGD optimizer, a small number of samples are randomly chosen—instead of using the entire dataset for each iteration and the torch.optim package is used. The learning rate was fixed at 0.001.
The exponential learning rate decays the learning rate of each parameter group by gamma every epoch. This means that the learning rate will decrease rapidly in the first few epochs, and spend more epochs with a lower value, but never reach exactly zero.

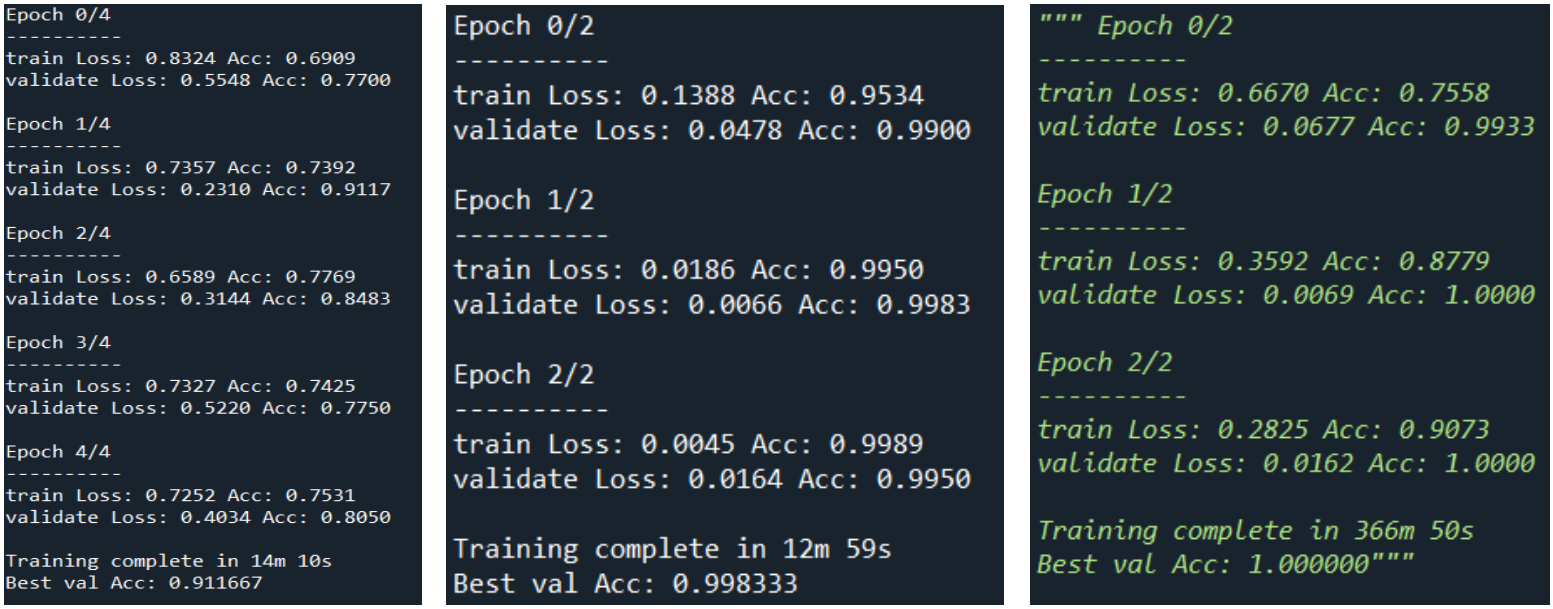
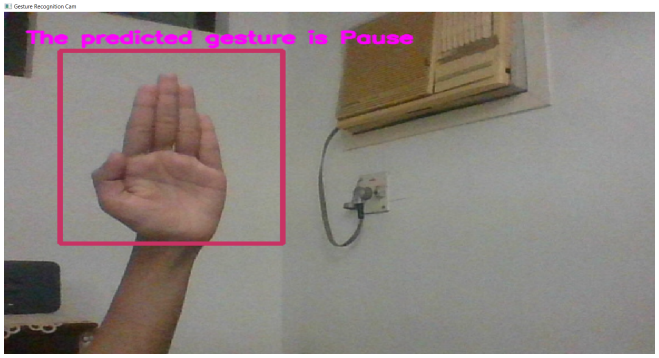


Fig 8. Iteration of each model over the number of epochs to get accuracy values- ResNet, AlexNet, and VGG16

The AlexNet model was saved as a .pt file which would be loaded for testing.

Stage 3 - Model Testing

The saved model is then loaded. The webcam is opened with a rectangular frame similar to the image capturing process. The different gestures are performed on the frame and the model will predict the gestures and display its label correctly on the webcam stream.



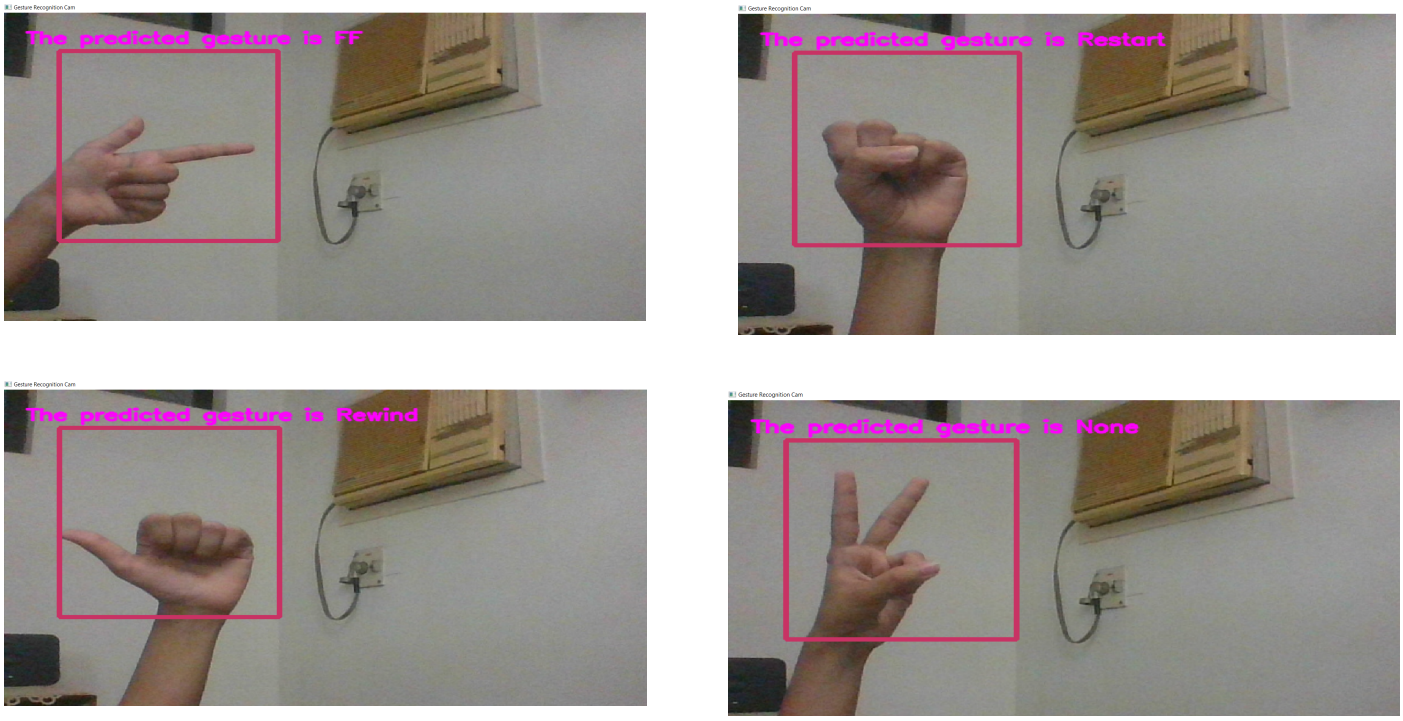


Fig 9 . Predicted hand gestures in webcam for model testing

As seen in Fig 9., each of the gestures is predicted accurately when the hand is placed inside the rectangular frame. The None label comes up when it is a plain background with no gestures present or when another gesture apart from the 5 recorded ones is placed in the frame.The AlexNet model was able to predict all the gestures correctly.

V. OBSERVATION

Confusion matrices were built on all the models to evaluate their performances.

ResNet

```
print(confusion_matrix.diag()/confusion_matrix.sum(1))
tensor([1.0000, 0.9400, 1.0000, 0.6100, 0.6800, 1.0000])
```

AlexNet

```
print(confusion_matrix.diag()/confusion_matrix.sum(1))
tensor([1.0000, 0.9800, 1.0000, 0.9900, 0.9400, 0.9900])
```

Name of the model	Best Validation Accuracy	Best Training Accuracy	Number of epochs
ResNet18	91.1%	77.6%	5
AlexNet	99.8%	99.8%	3
VGG16	100%	90.7%	3

After the performance of each model was evaluated, AlexNet emerged to be the best working model by looking at the training and validation accuracy as well as their confusion matrix.VGG16 also gave a high accuracy but the computational time was very lengthy.AlexNet had the least difference in the training and validation accuracy and also achieved a high accuracy.

VI. CONCLUSION

The hand gesture recognition system for managing media players uses real-time user input via an embedded webcam and provides gesture matches with a function to control the media player. The objective of this project was to create a recognition system by loading and using pretrained models for classifying the different gestures and predicting them in real time. The three pretrained models were loaded through transfer learning after which we were able to evaluate their performances and save and load the model giving the most optimal accuracy values. Developing a system like this makes the interaction between humans and computers more natural and the concept can be used for various applications as in a media player in our case.

VII. FUTURE SCOPE

Media player has become one of the fundamental parts of our daily lives. The face and hand gesture recognition system for controlling the media player works using real time gesture input from users using an integrated webcam and provides gesture matches with a function to control the media player.

Widespread acclaim has been accorded to gesture recognition as a result of technological breakthroughs that allow unique, easy, and speedy means of human-computer interaction. Existing systems offer excellent functionality but have not been readily welcomed by consumers. The primary issue with these systems is their poor accuracy and complicated algorithms. The suggested system will seek to address these problems and distinguish itself from other gesture recognition systems. It will offer a non-touch interface for manipulating multimedia files and programs, such as video and music players. It will serve as a system manipulation assistance for those with impairments, those who cannot use their input devices, and those who prefer this more natural mode of communication over others.

Potentially, gesture control for media players might replace computer hardware such as the mouse and keyboard. Consequently, the price of computer hardware may decrease. A substantial amount of e-waste is created by computer gear. This program may decrease the amount of e-waste created by certain hardware components.

However, the effectiveness of a vision-based gesture interaction is susceptible to being affected by changes in lighting, complex backgrounds, camera movement, and particular user variance.

VIII. REFERENCES

https://pytorch.org/vision/stable/transforms.html

https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html

https://pytorch.org/docs/stable/optim.html

https://pytorch.org/docs/stable/nn.html

https://pytorch.org/docs/stable/nn.functional.html

https://link.springer.com/article/10.1007/s41060-016-0008-z#Sec4

http://www.diva-portal.org/smash/get/diva2:1299000/FULLTEXT01.pdf

https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

https://towardsdatascience.com/training-a-neural-network-to-detect-gestures-with-opencv-in-python-e09b0a12bdf1

https://www.researchgate.net/publication/354303600_Face_and_Hand_Gesture_Recognition_System_for_Controlling_VLC_Media_Player

****