OBJECT CLASSIFICATION FROM VIDEO WITH CNN

PROJECT REPORT

Submitted by

AVIJIT SEN (12621010008)

SNEHA DAS (12621010041)

SWAGATA DE (12621010055)

SAMARPITA SAHA (12621010034)

SUMAN MANDAL (12621010054)

in partial fulfilment for 3rd Semester Minor Project of

MASTER OF COMPUTER APPLICATION

DEPARTMENT OF COMPUTER APPLICATIONS

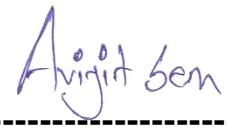
HERITAGE INSTITUTE OF TECHNOLOGY, KOLKATA

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY,WEST BENGAL,

DECEMBER, 2022

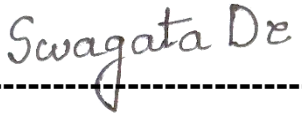
DECLARATION BY STUDENTS

This is to declare that this report "Object Classification with CNN" has been written by us. No part of the report is plagiarized from other sources. All information included from other sources have been duly acknowledged. We confirm that if any part of the report is found to be plagiarized, we shall take full responsibility for it.

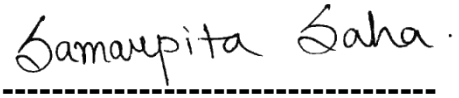


Avijit Sen
Roll No: 12621010008

Sneha Das
Roll No: 12621010041



Swagata De
Roll No: 12621010055

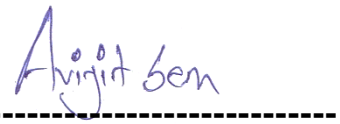


Samarpita Saha
Roll No: 12621010034

Suman Mandal
Roll No: 12621010054

ACKNOWLEDGEMENT

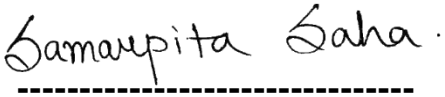
We are highly indebted to our project guide Prof. Anirban Kundu for guiding us and providing constant supervision and necessary information regarding the project & also for support in completing the project. We are also thankful to the Department of Computer Applications for providing us easy access to the project laboratory and other facilities.



Avijit Sen
Roll No: 12621010008

Sneha Das
Roll No: 12621010041

Swagata De
Roll No: 12621010055



Samarpita Saha
Roll No: 12621010034

Suman Mandal
Roll No: 12621010054

ABSTRACT

In recent years, there has been a rapid development in web users and sufficient bandwidth. Internet connectivity, which is so low cost, makes the sharing of information (text, audio and videos) more common and faster. This video content needs to be analysed for prediction its class in different purpose for the users. Many machines learning approach has been developed for the classification of video to save people time and energy. There are a lot existing review projects on video classification, but they have some limitations such as limitation of analysis, badly structured, not mention research gaps or findings, not clearly describe advantages, disadvantages, and future work. But our review project almost overcomes these limitations. This study attempts to review existing video-classification procedures and to examine the existing methods of video-classification comparatively and critically and to recommend the most effective and productive process. First of all, our analysis examines the classification of videos with taxonomical details, latest application, process and datasets information. Secondly, overall inconvenience, difficulties, shortcomings and potential work, data, performance measurements with the related recent relation in science, deep learning and the model of machine learning. Study on video classification systems using their tools, benefits, drawbacks, as well as other features to compare the techniques they have used also constitutes a key task of this review. Lastly, we also present a quick summary table based on selected features. In terms of precision and independence extraction functions, the RNN (Recurrent Neural Network), CNN (Convolutional Neural Network) and combination approach performs better than the CNN dependent method.

TABLE OF CONTENTS

- OVERVIEW
- ENVIRONMENT SETUP
- ARTIFICIAL INTELLIGENCE
- MACHINE LEARNING
- DEEP LEARNING
    - ARTIFICIAL NEURAL NETWORK
    - RECURRENT NEURAL NETWORK
    - CONVOLUTIONAL NEURAL NETWORK
- OBJECT CLASSIFICATION FROM VIDEO WITH CNN
    - INTRODUCTION
    - DETAILS OF PROJECT
        - DATA COLLECTION
        - ABOUT THE DATA
    - SYSTEM REQUIREMENTS
        - SOFTWARE REQUIREMENTS
        - HARDWARE REQUIREMENTS
- MODULES USED IN PROJECTS
    - NUMPY
    - CV2
    - OS
    - MATPLOTLIB.PYPLOT
    - TENSORFLOW
    - GOOGLE.COLAB
    - KERAS
- CODES WITH OUTPUT
- CONCLUSION
- FUTURE SCOPES
- BIBLIOGRAPHY

➢ OVERVIEW

The internet is currently commonly used by the people worldwide. Social media have an essential role to play of content distribution (audio, video, text, image) sharing. About the same period, they also share their emotions in social media about a certain aspect so that those users can quickly find out exactly what is happening and with this reason, user views are used to estimate the public opinion on certain issues. However, if consumer employ a person to evaluate the views of people through multitudes of content it is very difficult and time consuming. In order to evaluate public attitudes, the researchers present a machine learning approach to data mining. Video classification is part of mining which analyses text through natural language processing, video by machine linguistics in order to find views of people by gathering and analysing social and other resources of subjective knowledge. Deep learning methodology is more reliable and effective than other approaches.

This project is on only deep learning-based approach for video classification with good description on deep model, data and feature extraction tools but does not able to mention research gaps, advantages and performance. Human Behaviour Analysis (HBA) is a significant field of focus in artificial intelligence. It has many fields of use such as video monitoring, environment-assisted living, smart shopping environments, etc. The availability of human video data is increasing dramatically, with the help of leading companies in this area. From the perspective of DL this job approaches HBA. Owing to the growth of computing capacity, Deep Learning techniques have been a great step in the classification context in the last few years. Strategies used are Convolutional Neural Networks (CNNs) for image comprehension, and RNNs for temporary comprehension like video or text.

- **Objectives of the System:**

  The purpose of the system is to develop a transparent, globally accessible and immutable records of resource availability and need. The primary objective of the proposed system is to provide decision support to suppliers

for resource allocation in a proper way. The main objectives may be summed up as follows:

- Determining the exact demand for different emergency resources at everyshelter.
- Enumerate the exact utility of different emergency resources at everyshelter.
- Prevent improper resources allocation and black marketing.

- **Design Goals of the System:**
  - Demand Forecasting with Principal Component Regression Model.
  - Utility Enumeration through Utility Function.
  - Optimal Resource Allocation by Utility Based Integer Programming Model.
  - Block chain based immutable and globally accessible record generation for resource requirement vis-à-vis availability
- **Users of the System:**
  - Volunteers - Personnel in charge of a particular shelter and are affiliated to a particular control station.
  - Command and control station – It centralizes the monitoring, control and command of the system's overall operations.
  - Miners – The people who verify and store the record of resource availability and need.
- **Software requirement Specification:**

  A Software Requirements Specification (SRS) is a comprehensive description of the environment for a system under development. The SRS fully describes what the system will do and how it is expected to perform.
- **Functional Requirements:**
  - Registered volunteers are supposed to upload the values of the Influencing Situational Parameters (ISPs) every day. They can change theiruploaded values within their accessible time for that day only.

- Registered minors are supposed to verify the values of the influencing parameters and can block or delete the volunteer accounts from the system if they are found to be malicious.
- Registered minors are able to calculate the demand and utilityvalues and store them for a particular day.
- Command and control station is able to view the new demand and utility values and allocate the resources for a particular shelter accordingly.

- **Non-Functional Requirements:**

  The system should have the following characteristics:

  - **Usable:** The interface should be intuitive and User-friendly.
  - **Available:** The system should be available to volunteers andcontrol station to 6am-6pm and to minors to 6pm-6am (next day)
  - **Secure:** The system should be protected against unauthorizedaccess. User confidentiality should be ensured.
  - **Reliable:** The Mean-Time-Between-Failures and Mean-Time-To-Recovery metrics should be as low as possible.
  - **Maintenance of data integrity:** Data integrity-such as referential integrity of the system's database should be maintained at all times.

- **Hardware and Software Requirement Specifications:**
  - **Hardware Requirements:**
    - **Processor:** CPU @ 2.0 GHz
    - **RAM:** Minimum 1 GB.
    - **Storage:** Minimum 160 GB.
  - **Software Requirements:**
    - **Operating System**: Windows XP and Later.
    - **Programming Language**: Python
    - **Documentation:** Microsoft office Word.

➢ ENVIRONMENT SETUP

Open a terminal window and type "python" to find out if it is alreadyinstalled and which version is installed.

- UNIX (Solaris, Linux, FreeBSD, AIX, HP/UX, SunOS, IRIX, etc.)
- Win 9x/NT/2000
- Macintosh (Intel, PPC, 68K)
- OS/2
- DOS (multiple versions)
- PalmOS
- Windows CE
- Acorn/RISC OS

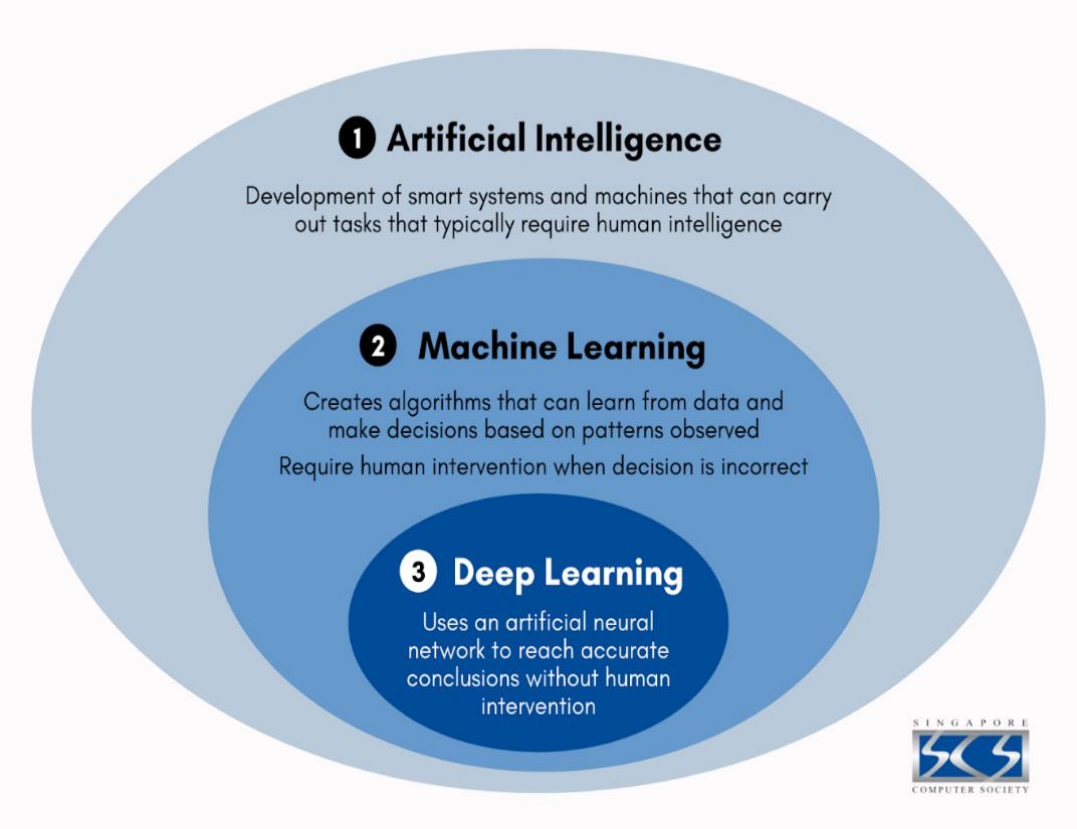


fig: Overview of Artificial Intelligence, Machine and Deep Learning

➢ ARTIFICIAL INTELLIGENCE

According to the father of Artificial Intelligence, John McCarthy, it is "The science and engineering of making intelligent machines, especially intelligent computer programs".

Artificial Intelligence is a way of making a computer, a- computer-controlledrobot, or a software think intelligently, in the similar manner the-intelligent humans think.AI is accomplished by studying how the human brain thinks, and how humanslearn, decide, and work while trying to solve a problem, and then using theoutcomes of this study as a basis-of developing intelligent software and systems. The development of AI started with the intention of creating similar intelligence in machines that we find and regard high in humans.

- **Goals of AI**
  - **To Create Expert Systems:** The systems which exhibit intelligent behaviour,learn, demonstrate, explain, and advise its users.
  - **To Implement Human Intelligence in Machines:** Creating systems that understand, think, learn, and behave like humans.
- **Applications of AI**

  AI has been dominant in various fields such as Gaming: AI plays a crucial role in strategic games such as chess, poker, tic-tac-toe, etc., where machines can think of a large number of possiblepositions based on heuristic knowledge.
  - **Natural Language Processing:** It is possible to interact with a computer thatunderstands natural language spoken by humans.
  - **Expert Systems:** There are some applications which integrate machine, software, andspecial information to impart reasoning and advising. They provide explanations and advice to the users.
  - **Vision Systems:** These systems understand, interpret, and comprehend visual inputon the computer.

    For example: A spying aero plane takes photographs, which are used to figure out spatial information or map of the areas. Police use computer software that can recognize the face of a criminal with the storedportrait made by a forensic artist.
  - **Speech Recognition:** Some intelligent systems are capable of hearing and comprehending the language in terms of sentences and their meanings while a human talk to it. It can handledifferent accents, slang words, noise in the background, change in human's noise due to cold, etc.

➢ MACHINE LEARNING

Machine Learning, as the name says, is all about machines learning automatically without being explicitly programmed or learning without any direct human intervention. This machine learning process starts with feeding them good quality data and then training the machines by building various machine learning models using the data and different algorithms. The choice of algorithms depends on what type of data we have and what kind of task we are trying to automate.

As for the formal definition of Machine Learning, we can say that a Machine Learning algorithm learns from experience

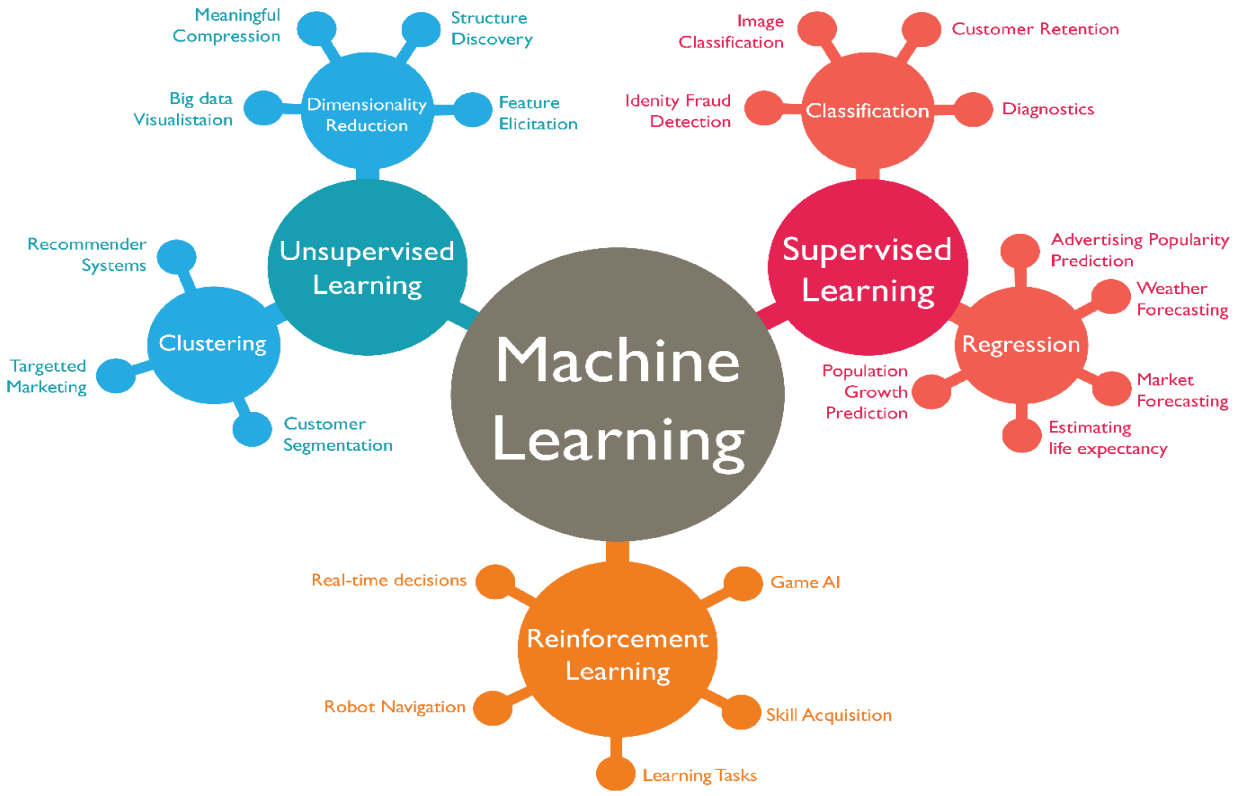


fig: Machine Learning & It's Types

E with respect to some type of task T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

For example, if a Machine Learning algorithm is used to play chess. Then the experience E is playing many games of chess, the task T is playing chess with many

players, and the performance measure P is the probability that the algorithm will win in the game of chess.

ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect. Machine is an essential skill for any aspiring data analyst and data scientist, and also for those who wish to transform a massive amount of raw data into trends and predictions. Learn this skill today with Machine Learning Foundation – Self Paced Course , designed and curated by industry experts having years of expertise in ML and industry-based projects.

➢ DEEP LEARNING

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labelled data and neural network architectures that contain many layers.

In a word, accuracy. Deep learning achieves recognition accuracy at higher levels than ever before. Recent advances in deep learning have improved to the point where deep learning outperforms humans in some tasks like classifying objects in images.

Deep learning models are trained by using large sets of labelled data and neural network architectures that learn features directly from the data without the need for manual feature extraction. a set of interconnected nodes. Networks can have tens or hundreds of hidden layers.

- TYPES OF DEEP LEARNING:
  - ❖ ARTIFICIAL NEURAL NETWORK
  - ❖ RECURRENT NEURAL NETWORK
  - ❖ CONVOLUTIONAL NEURAL NETWORK

❖ **ARTIFICIAL NEURAL NETWORK (ANN):**

Artificial Neural Network is a group of multiple perceptron or neurons at each layer. Artificial Neural Network is also known as a Feed-Forward Neural network. This type of neural networks are one of the simplest variants of neural networks. They pass information in one direction, through various input nodes, until it makes it to the output node. The network may or may not have hidden node layers, making their functioning more interpretable.

Storing information on the entire network.

- Ability to work with incomplete knowledge.
- Having fault tolerance.
- Having a distributed memory.

❖ **Recurrent Neural Network (RNN)**:

Recurrent Neural Network is a type of Neural Network where the output from the previous step is fed as input to the current step. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus, RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is Hidden state, which remembers some information about a sequence.

RNN have a "memory" which remembers all information about what has been calculated. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

Now the RNN will do the following:

- RNN converts the independent activations into dependent activations by providing the same weights and biases to all the layers, thus reducing the complexity of increasing parameters and memorizing each previous output by giving each output as input to the next hidden layer.
- Hence these three layers can be joined together such that the weights and bias of all the hidden layers are the same, in a single recurrent layer.

❖ **CONVOLUTION NEURAL NETWORK (CNN):**

One of the most popular types of deep neural networks is known as convolutional neural networks (CNN or ConvNet). A CNN convolves

learned features with input data, and uses 2D convolutional layers, making this architecture well suited to processing 2D data, such as images.

CNNs eliminate the need for manual feature extraction, so you do not need to identify features used to classify images. The CNN works by extracting features directly from images. The relevant features are not pretrained; they are learned while the network trains on a collection of images. This automated feature extraction makes deep learning models highly accurate for computer vision tasks such as object classification. The output of each convolved image serves as the input to the next layer.

CNNs learn to detect different features of an image using tens or hundreds of hidden layers. Every hidden layer increases the complexity of the learned image features. For example, the first hidden layer could learn how to detect edges, and the last learns how to detect more complex shapes specifically catered to the shape of the object we are trying to recognize.

- Very High accuracy in image recognition problems.
- Automatically detects the important features without any human supervision

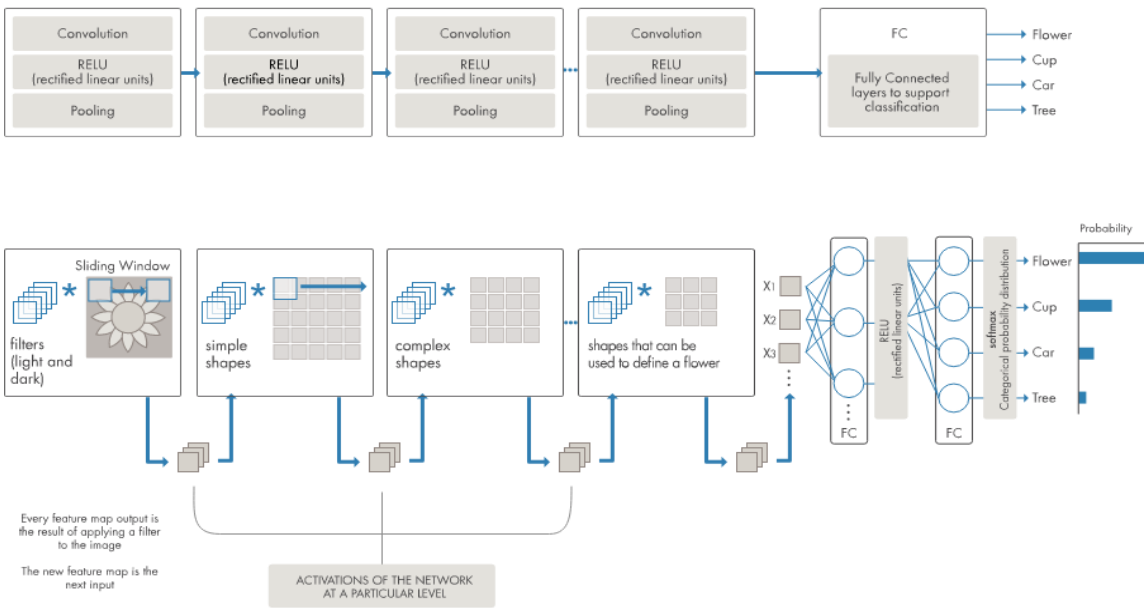


fig: Overview of CNN

- Convolutional Neural Network Architecture

A CNN typically has three layers: a convolutional layer, a pooling layer, and a fully connected layer.

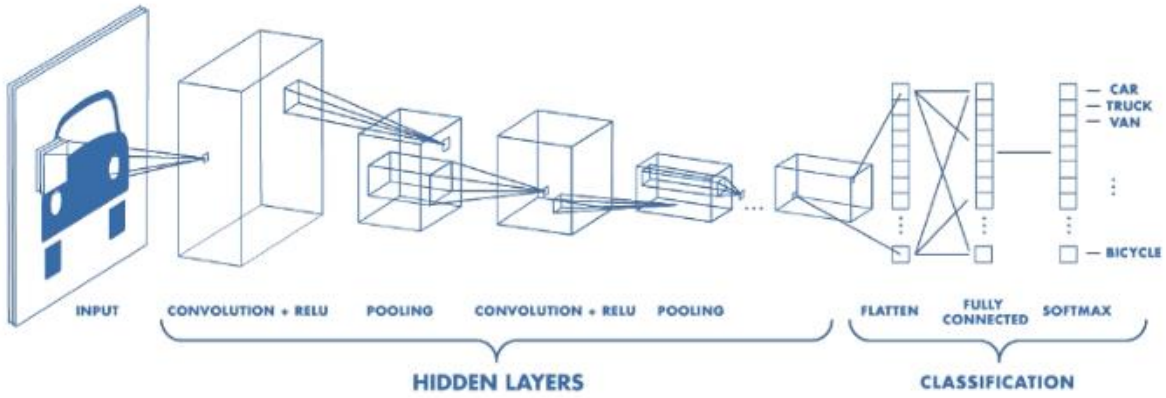


fig: Architecture of CNN

- Convolution Layer

The convolution layer is the core building block of the CNN. It carries the main portion of the network's computational load.

This layer performs a dot product between two matrices, where one matrix is the set of learnable parameters otherwise known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image but is more in-depth. This means that, if the image is composed of three (RGB) channels, the kernel height and width will be spatially small, but the depth extends up to all three channels.

During the forward pass, the kernel slides across the height and width of the image-producing the image representation of that receptive region. This produces a two-dimensional representation of the image known as an activation map that gives the response of the kernel at each spatial position of the image. The sliding size of the kernel is called a stride.

If we have an input of size W x W x D and Dout number of kernels with a spatial size of F with stride S and amount of padding P, then the size of output volume can be determined by the following formula:

$$W_{out} = \frac{W - F + 2P}{S} + 1$$

This will yield an output volume of size Wout x Wout x Dout.

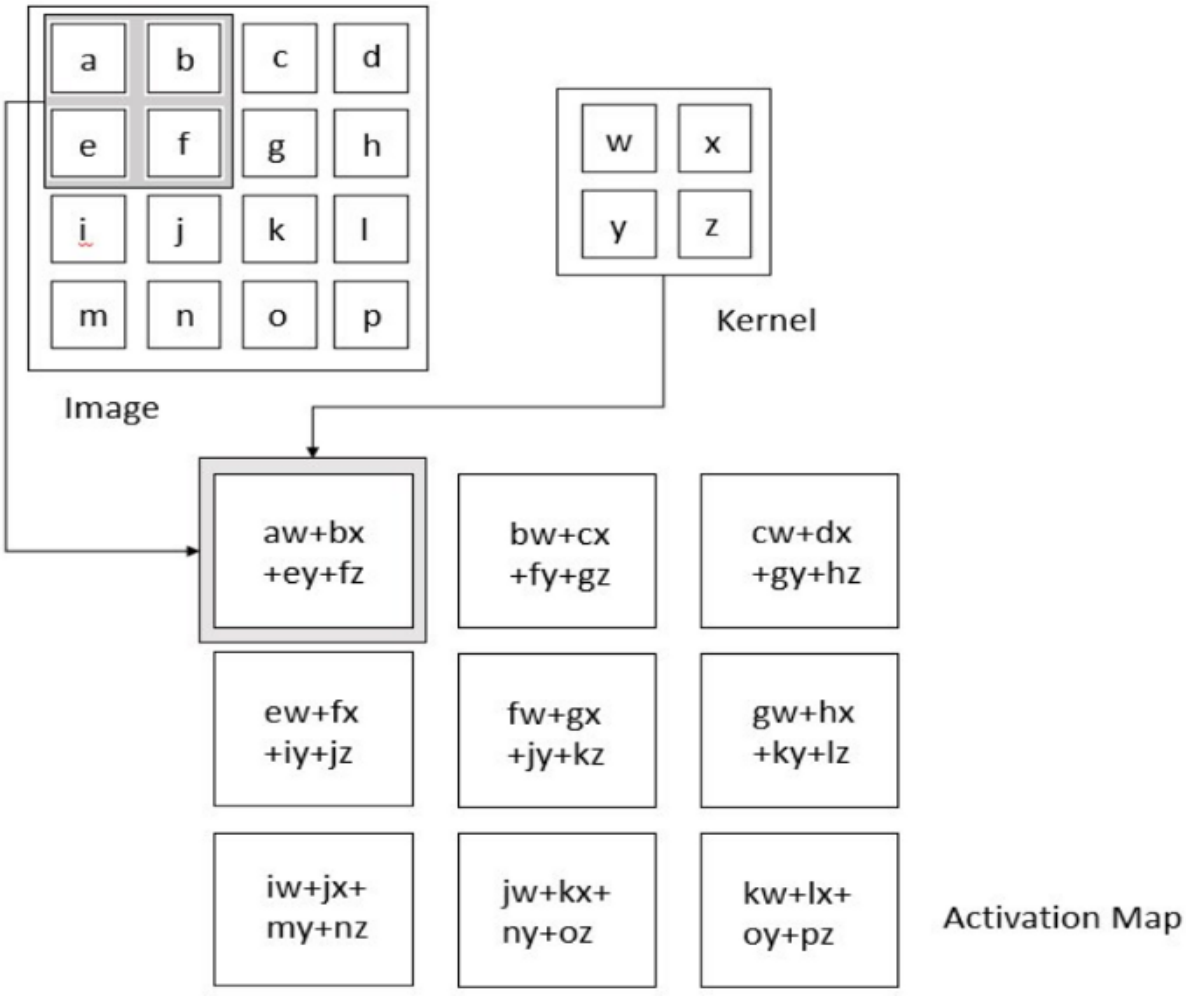


fig: Convolutional Operation

- Motivation behind Convolution

  Convolution leverages three important ideas that motivated computer vision researchers: sparse interaction, parameter sharing, and equivariant representation. Let's describe each one of them in detail.

  Trivial neural network layers use matrix multiplication by a matrix of parameters describing the interaction between the input and output unit. This means that every output unit interacts with every input unit. However, convolution neural networks have sparse interaction. This is achieved by making kernel smaller than the input e.g., an image can have millions or thousands of pixels, but while processing it using kernel, we can detect meaningful information that is of tens or hundreds of pixels. This means that we need to store fewer parameters that not only reduces the memory requirement of the model but also improves the statistical efficiency of the model.

  If computing one feature at a spatial point (x1, y1) is useful then it should also be useful at some other spatial point say (x2, y2). It means that for a single two-dimensional slice i.e., for creating one activation map, neurons are constrained to use the same set of weights. In a traditional neural network, each element of the weight matrix is used once and then never revisited, while convolution network has shared parameters i.e., for getting output, weights applied to one input are the same as the weight applied elsewhere.

  Due to parameter sharing, the layers of convolution neural network will have a property of equivariance to translation. It says that if we changed the input in a way, the output will also get changed in the same way.

- Pooling Layer

  The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing

the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

There are several pooling functions such as the average of the rectangular neighbourhood, L2 norm of the rectangular neighbourhood, and a weighted average based on the distance from the central pixel. However, the most popular process is max pooling, which reports the maximum output from the neighbourhood.

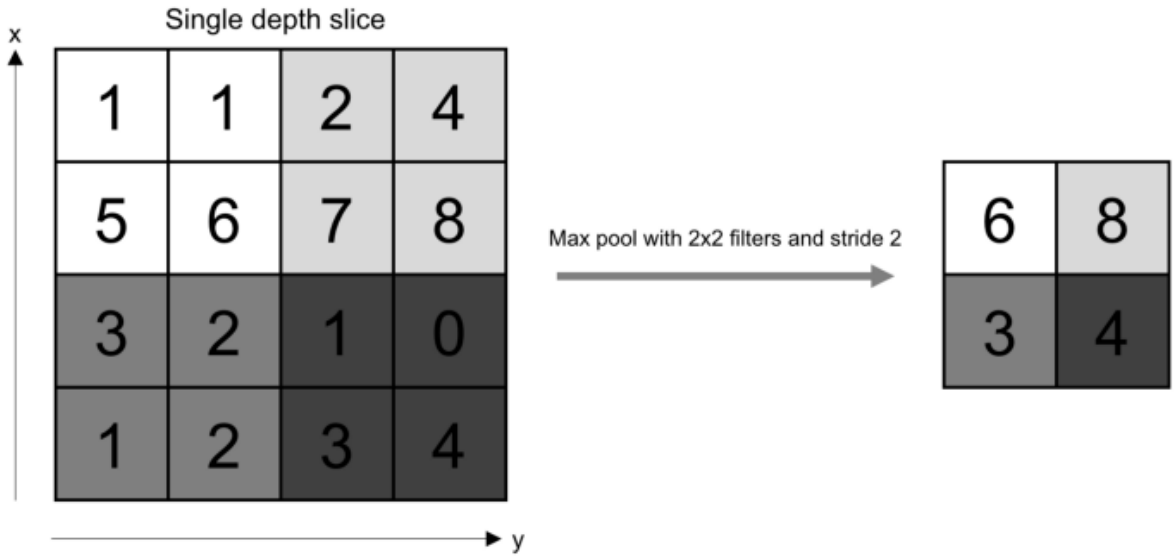


fig: Max Pooling Operation

If we have an activation map of size W x W x D, a pooling kernel of spatial size F, and stride S, then the size of output volume can be determined by the following formula:

$$W_{out} = \frac{W - F}{S} + 1$$

This will yield an output volume of size Wout x Wout x D.

In all cases, pooling provides some translation invariance which means that an object would be recognizable regardless of where it appears on the frame.

- Fully Connected Layer

  Neurons in this layer have full connectivity with all neurons in the preceding and succeeding layer as seen in regular FCNN. This is why it can be computed as usual by a matrix multiplication followed by a bias effect.

  The FC layer helps to map the representation between the input and the output.

- Non-Linearity Layers

  Since convolution is a linear operation and images are far from linear, non-linearity layers are often placed directly after the convolutional layer to introduce non-linearity to the activation map.

  There are several types of non-linear operations, the popular ones being:

  - **Sigmoid:** The sigmoid non-linearity has the mathematical form $\sigma(\kappa) = 1/(1+e^{-}\kappa)$. It takes a real-valued number and "squashes" it into a range between 0 and 1. However, a very undesirable property of sigmoid is that when the activation is at either tail, the gradient becomes almost zero. If the local gradient becomes very small, then in backpropagation it will effectively "kill" the gradient. Also, if the data coming into the neuron is always positive, then the output of sigmoid will be either all positives or all negatives, resulting in a zig-zag dynamic of gradient updates for weight.
  - **Tanh:** Tanh squashes a real-valued number to the range [-1, 1]. Like sigmoid, the activation saturates, but — unlike the sigmoid neurons its output is zero centred.
  - **ReLU:** The Rectified Linear Unit (ReLU) has become very popular in the last few years. It computes the function $f(\kappa)=\max(0, \kappa)$. In other words, the activation is simply threshold at zero. In comparison to sigmoid and tanh, ReLU is more reliable and accelerates the convergence by six times.

➢ OBJECT CLASSIFICATION FROM VIDEO WITH CNN

- **Challenges in Video classification:**

The field of computer vision has experienced substantial progress recently, owing largely to advances in deep learning, specifically convolutional neural nets (CNNs). Image classification, where a computer classifies or assigns labels to an image based on its content, can often see great results simply by leveraging pre-trained neural nets and fine-tuning the last few throughput layers. Classifying *and* finding an unknown number of individual objects within an image, however, was considered an extremely difficult problem only a few years ago. This task, called object detection, is now feasible and has even been productized by companies like Google and IBM. But all of this progress wasn't easy! Object detection presents many sizable challenges beyond what is required for image classification. After a brief introduction to the topic, let's take a deep dive into several of the interesting obstacles these problems pose along with various emerging solutions.

❖ INTRODUCTION

- What is object classification from video?

  Object classification is part of mining which analyses text through natural language processing, video by machine linguistics in order to find views of people by gathering and analysing social and other resources of subjective knowledge. Deep learning methodology is more reliable and effective than other approaches. This project is on only deep learning-based approach for video classification with good description on deep model, data and feature extraction tools but does not able to mention research gaps, advantages and performance. Human Behaviour Analysis (HBA) is a significant field of focus in artificial intelligence. It has many fields of use such as video monitoring, environment-assisted living, smart shopping environments, etc. The availability of human video.

  Data is increasing dramatically, with the help of leading companies in this area. From the perspective of DL this job approaches HBA. Owing to the growth of computing capacity, Deep Learning techniques have been a great step in the classification context in the last few years. Strategies used are Convolutional Neural Networks (CNNs) for image comprehension, and RNNs for temporary comprehension like video or text.

❖ DETAILS OF THE PROJECT

- **Data Collection:** Data is collected from our photos and videos that is uploaded in Drive.
- **About the Data:** Object recognition is an expansive research area that already contains detailed ways of implementation which include major learning datasets, popular algorithms, features scaling and feature extraction methods.

❖ SYSTEM REQUIREMENTS

- **SOFTWARE REQUIREMENTS:**
  - Os: windows or Linux
  - Python-IDE: python2.7x and above
  - Colab Notebook
  - Setup tools and pin be installed for 3.6 and above
  - Language python
- **HARDWARE REQUIREMENTS:**
  - RAM:4GB and higher
  - Processor: Intel i3 and above
  - Hard disk:500GB: minimum

➢ MODULES USED IN THIS PROJECT

- **OS:** The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The *Os* and *Os. Path* modules include many functions to interact with the file system.
- **CV2:** Cv2 is the module import name for OpenCV-python, "Unofficial pre-built CPU-only OpenCV packages for Python". The traditional OpenCV has many complicated steps involving building the module from scratch, which is unnecessary. I would recommend remaining with the OpenCV-python library.
- **NumPy:** It is a general-purpose array processing package.it provides a high-performance multidimensional array object, and tools for working with these arrays. it's also an efficient multidimensional container of generic data.
- **Matplotlib:** It is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.it tries to make easy things easy and hard things possible. you can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc., with just a few lines of code.
- **TensorFlow:** TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow.

  Tensors are just buckets of numbers of a specific shape and a certain rank (dimensionality). Tensors are used in Machine Learning with TensorFlow to represent input data and output data (and everything in between) in Machine Learning models. A tensor can be described as a n-dimensional numerical array. A tensor can be called a generalized matrix. It could be a 0-D matrix (a single number), 1-D matrix (a vector), 2-D matrix or any higher dimensional structure. A tensor is identified by three parameters viz., rank, shape and size.

- **Keras**: Keras is a neural network Application Programming Interface (API) for Python that is tightly integrated with TensorFlow, which is used to build machine learning models. Keras models offer a simple, user-friendly way to define a neural network, which will then be built for you by TensorFlow.
- **Colab:** Google Colab is a free Jupiter notebook that allows to run Python in the browser without the need for complex configuration. It comes with Python installed and has all the main Python libraries installed. It also comes integrated with free GPUs.

➢ CODE WITH OUTPUT

- Modules Used
- Data Collection
- Frame Extracting
- Model Selection
- Training
- Testing

- **Modules Used:** Here we import python in built modules to implement our model.

```
# import all modules
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
from tensorflow.keras.optimizers import RMSprop
from google.colab import drive
import tensorflow as tf
import os
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

- **Data Collection:** Data is uploaded in drive and used as Training and Validation datasets. Here two datasets are Avijit and Swagata which contain their photos respectively.

```
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
# Rescaling the images
train = ImageDataGenerator(rescale= 1/255)
validation = ImageDataGenerator(rescale= 1/255)
```

```
train_dataset = train.flow_from_directory('/content/drive/MyDrive/Project_Classification/Training/',
                                          target_size= (200,200),
                                          batch_size= 3,
                                          class_mode= 'binary')
```

```
Found 140 images belonging to 2 classes.
```

```
validation_dataset =train.flow_from_directory('/content/drive/MyDrive/Project_Classification/Validation/' ,
                                              target_size= (200,200),
                                              batch_size= 3,
                                              class_mode= 'binary')
```

```
Found 40 images belonging to 2 classes.
```

```
train_dataset.class_indices
```

```
{'Avijit': 0, 'Swagata': 1}
```

```
train_dataset.classes # Values for Each Images in the Folder
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1], dtype=int32)
```

- **Frame Extracting from Video:** Here we extract the frames from uploaded video to form Testing dataset.

```
vid = cv2.VideoCapture("/content/drive/MyDrive/Project_Classification/video1_FipjXWSF.mp4")
# exception handling
try:
    if not os.path.exists('/content/drive/MyDrive/Project_Classification/Testing_2'):
        os.makedirs('/content/drive/MyDrive/Project_Classification/Testing_2')
# if not created then raise error
except OSError:
    print('Error: Creating directory of data')
currentframe = 0
while (True):
    success, frame = vid.read()
    if success:
        # continue creating images until video remains
        name = '/content/drive/MyDrive/Project_Classification/Testing_2/' + str(currentframe) + '.jpg'
        print('Creating...' + name)
        # writing the extracted images
        cv2.imwrite(name, frame)
        # increasing counter so that it will
        # show how many frames are created
        currentframe += 1
    else:
        break
# Release all space and windows once done
vid.release()
cv2.destroyAllWindows()
```

Some output examples how the frames are extracting from video

```
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/0.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/1.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/2.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/3.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/4.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/5.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/6.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/7.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/8.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/9.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/10.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/11.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/12.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/13.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/14.jpg
Creating.../content/drive/MyDrive/Project_Classification/Testing_2/15.jpg
```

.....

Total 373 frames are extracted from the video.

- **Model Processing:**

```
# Algoritham of CNN
model = tf.keras.models.Sequential([ tf.keras.layers.Conv2D(16,(3,3),activation='relu',input_shape=(200,200,3)),
                                     tf.keras.layers.MaxPool2D(2,2),
                                     # Using Convolution Layer ,ReLU Activation function,Pooling layer
                                     tf.keras.layers.Conv2D(32,(3,3),activation='relu'),
                                     tf.keras.layers.MaxPool2D(2,2),
                                     # Again using Convolution Layer ,ReLU Activation function,Pooling layer
                                     tf.keras.layers.Conv2D(64,(3,3),activation='relu'),
                                     tf.keras.layers.MaxPool2D(2,2),
                                     # Again using Convolution Layer ,ReLU Activation function,Pooling layer
                                     tf.keras.layers.Flatten(),
                                     # Using fully connected layer
                                     tf.keras.layers.Dense(512,activation='relu'),
                                     tf.keras.layers.Dense(1,activation='sigmoid')])
```

```
model.compile(loss = 'binary_crossentropy',
              optimizer = RMSprop(lr=0.003),
              metrics = ['accuracy'])
```

- **Training:**

```
model_fit = model.fit(train_dataset,
                      steps_per_epoch = 3,
                      epochs = 30,
                      validation_data = validation_dataset)
```

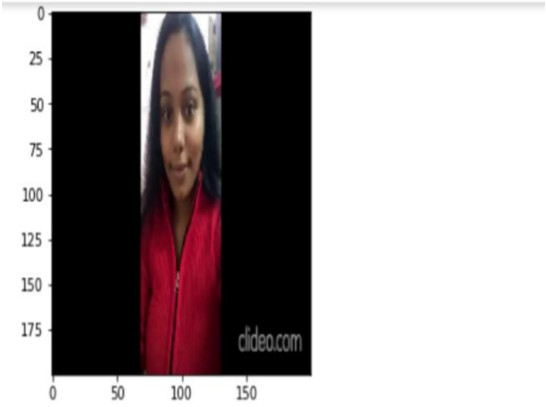
```
Epoch 1/30
3/3 [==============================] - 22s 10s/step - loss: 28.7561 - accuracy: 0.5556 - val_loss: 3.6941 - val_accuracy: 0.5000
Epoch 2/30
3/3 [==============================] - 3s 1s/step - loss: 1.8873 - accuracy: 0.4444 - val_loss: 0.6970 - val_accuracy: 0.5000
Epoch 3/30
3/3 [==============================] - 3s 1s/step - loss: 0.7784 - accuracy: 0.6667 - val_loss: 0.6843 - val_accuracy: 0.5250
Epoch 4/30
3/3 [==============================] - 3s 1s/step - loss: 0.5640 - accuracy: 0.8889 - val_loss: 1.1616 - val_accuracy: 0.5000
Epoch 5/30
3/3 [==============================] - 3s 1s/step - loss: 0.9297 - accuracy: 0.7778 - val_loss: 0.6690 - val_accuracy: 0.6000
Epoch 6/30
3/3 [==============================] - 3s 1s/step - loss: 0.3681 - accuracy: 0.8889 - val_loss: 0.7518 - val_accuracy: 0.5000
Epoch 7/30
3/3 [==============================] - 2s 989ms/step - loss: 0.8294 - accuracy: 0.6667 - val_loss: 3.1491 - val_accuracy: 0.5000
Epoch 8/30
3/3 [==============================] - 2s 926ms/step - loss: 0.0998 - accuracy: 1.0000 - val_loss: 2.2883 - val_accuracy: 0.5000
Epoch 9/30
3/3 [==============================] - 3s 1s/step - loss: 0.4752 - accuracy: 0.8889 - val_loss: 1.2900 - val_accuracy: 0.5000
.............
Epoch 17/30
3/3 [==============================] - 2s 957ms/step - loss: 0.9359 - accuracy: 0.8889 - val_loss: 0.5646 - val_accuracy: 0.6500
Epoch 18/30
3/3 [==============================] - 2s 949ms/step - loss: 0.4504 - accuracy: 0.7778 - val_loss: 0.6016 - val_accuracy: 0.6750
Epoch 19/30
3/3 [==============================] - 2s 1s/step - loss: 0.2405 - accuracy: 0.8889 - val_loss: 3.4870 - val_accuracy: 0.5000
Epoch 20/30
3/3 [==============================] - 2s 994ms/step - loss: 0.0161 - accuracy: 1.0000 - val_loss: 2.8411 - val_accuracy: 0.5000
Epoch 21/30
3/3 [==============================] - 2s 950ms/step - loss: 0.5700 - accuracy: 0.7778 - val_loss: 0.5669 - val_accuracy: 0.8000
Epoch 22/30
3/3 [==============================] - 2s 906ms/step - loss: 0.1174 - accuracy: 0.8889 - val_loss: 0.5484 - val_accuracy: 0.8000
Epoch 23/30
3/3 [==============================] - 2s 1s/step - loss: 0.1593 - accuracy: 0.8889 - val_loss: 0.5118 - val_accuracy: 0.8250
Epoch 24/30
3/3 [==============================] - 2s 952ms/step - loss: 0.1179 - accuracy: 1.0000 - val_loss: 1.6003 - val_accuracy: 0.5000
Epoch 25/30
3/3 [==============================] - 2s 958ms/step - loss: 0.2510 - accuracy: 0.7778 - val_loss: 0.6359 - val_accuracy: 0.6500
Epoch 26/30
3/3 [==============================] - 2s 924ms/step - loss: 0.3300 - accuracy: 0.7778 - val_loss: 0.5244 - val_accuracy: 0.8750
Epoch 27/30
3/3 [==============================] - 3s 1s/step - loss: 0.4011 - accuracy: 0.8889 - val_loss: 0.5520 - val_accuracy: 0.7750
Epoch 28/30
3/3 [==============================] - 2s 1s/step - loss: 0.0335 - accuracy: 1.0000 - val_loss: 0.6021 - val_accuracy: 0.5000
Epoch 29/30
3/3 [==============================] - 2s 978ms/step - loss: 3.8742e-04 - accuracy: 1.0000 - val_loss: 0.6080 - val_accuracy: 0.5000
Epoch 30/30
3/3 [==============================] - 2s 897ms/step - loss: 1.6088e-04 - accuracy: 1.0000 - val_loss: 0.6114 - val_accuracy: 0.5000
```

- **Testing:**

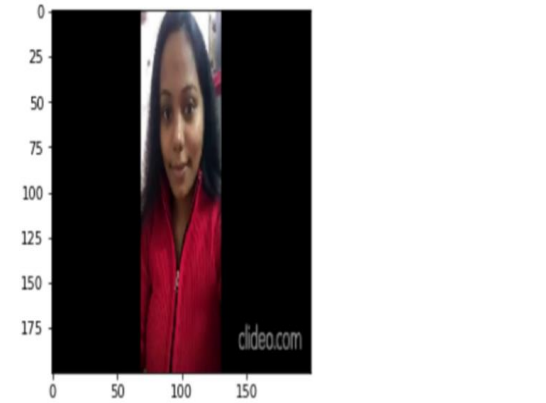
```
from tensorflow.keras.preprocessing import image
dir_path = '/content/drive/MyDrive/Project_Classification/Testing_2'
for i in os.listdir(dir_path):
 img2 = image.load_img(dir_path+'//'+ i,target_size=(200, 200))
 plt.imshow(img2)
 plt.show()
 X = image.img_to_array(img2)
 X = np.expand_dims(X,axis=0)
 image2 = np.vstack([X])
 val = model.predict(image2)
 if val == 1:
   print("Swagata")
 if val==0:
    print("Avijit")
```


```
1/1 [==============================] - 0s 36ms/step
Avijit
```


```
1/1 [==============================] - 0s 35ms/step
Avijit
```




```
1/1 [==============================] - 0s 35ms/step
Swagata
```




```
1/1 [==============================] - 0s 33ms/step
Swagata
```

Those 4 pictures are few samples of our output which is generated by our Model. Now our model can classify Avijit and Swagata successfully from the video.

➢ CONCLUSION

This article critically reviews on different approach and method in video classification with their advantage, finding, limitations, challenges, data summary, research gaps, and performance. From the analysis of this project. It is concluded that video-based approach for video classification works better over text and audio. The least employed process of video classification becomes text extraction. In different applications audio and video features extractions are used, but as we can appreciate, also the performance of the classification tasks can be more enhanced if the extraction both of visual and audio features is taken with same importance in the collection of video features. Audio-based solution needs little computing source. We also have the chance to identify videos in multiple ways to overcome the limitations of existing methods. By first segmenting images, we will identify them and then use the threshold and afterwards classify them in order to create new techniques. We may use movie and game or event forecasting classification algorithms. In order to identify the aspect of the movie and also the songs, fighting scene, funny scene, here is also a chance to classify videos. There are many existing methods for video classification and already have shown their performance. This review article also shows limitations of existing methods like unable to handle multiple features at a time, higher training time of deep learning, less adaptability of traditional machine learning, low accuracy to handle multilevel video. To overcome the limitations of the video classification is the trends and opportunity for the researcher. To classify longer video, to recognize multiple action in video, to find correlation among different videos, classification of multiple objects action in the video. Live steaming game video prediction is also trending and future work in video classification.

➢ FUTURE SCOPES

This research may be generalized to incorporate new methods in the future. However, the characteristics of frames as well as frame retrieval are key to the effective video classification. The role of patterns is also to boost the quality of the classification tasks. Another potential challenge of incorporating larger types of videos into the dataset with more efficient and generic functionality, to research methods that expressly clarify camera movement. To classify longer video, to recognize multiple action in video, to find correlation among different videos, classification of multiple objects action in the video. Live steaming game video prediction is the trends work in video classification.

➢ BIBLIOGRAPHY

- **Machine Learning from https://machinelearningmastery.com/object-recognition-convolutional-neural-networks-keras-deep-learning-library/**

- **Tensor flow from https://www.tensorflow.org/learn**

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

- **Documentation from https://www.mdpi.com/2079-9292/10/20/2470**