**BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI**

**Work Integrated Learning Programs Division**

**Post Graduate Program**

**in**

**Artificial Intelligence and Machine Learning**

**Video Analysis, Action & Event Recognition**

CAPSTONE PROJECT

Submitted in partial fulfillment of the requirements of the

Post Graduate Certification Program

in

Artificial Intelligence and Machine Learning

By

Rajeev Kumar 2020AIML004

Jaydeep Ghose 2020AIML016

Sai Satya Chandra Edara 2020AIML063

Under the supervision of

Gautam Gangopadhyay

Project work carried out at

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

Pilani (Rajasthan) INDIA

(Nov 2021)

PCAM ZC321 CAPSTONE PROJECT

**Video Analysis, Action & Event Recognition**

Submitted in partial fulfillment of the requirements of the

PGP - Artificial Intelligence and Machine Learning

By

(Rajeev Kumar, Jaydeep Ghose,

Sai Satya Chandra Edara)

(2020AIML004, 2020AIML016,

2020AIML063)

Under the supervision of

(Gautam Gangopadhyay)

Project work carried out at

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

PILANI (RAJASTHAN)

(Nov 2021)

**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI**

**CERTIFICATE**

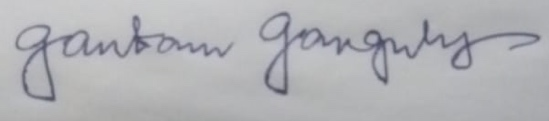
This is to certify that the Capstone Project entitled **Video Analysis, Action & Event Recognition**

and submitted by Mr. Rajeev Kumar, Mr. Jaydeep Ghose,

Mr. Sai Satya Chandra Edara ID No. 2020AIML004, 2020AIML016, 2020AIML063

in partial fulfillment of the requirements of PCAM ZC321 Capstone Project, embodies the work

done by him/her under my supervision.



Place: Kolkata Signature of the Mentor

Date: 10th November 2021 Name: GAUTHAM GANGOPADHYAY

**BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI**

**Work Integrated Learning Programs Division 2020-21**

**PCAM ZC321 CAPSTONE PROJECT**

|  |  |
| --- | --- |
|  | |
| Project Title : **Video Analysis, Action & Event Recognition**  Name of Mentor : **Gautam Gangopadhyay**  Name of Student : Rajeev Kumar, Jaydeep Ghose,  Sai Satya Chandra Edara  ID No. of Student : 2020AIML004, 2020AIML016,  2020AIML063 | Abstract This project is to automate the action recognition system based on Machine learning and Neural Network models to classify the video data. This can hence be used for automatic human activity recognition and event detection, in applications such as surveillance, entertainment and healthcare. |

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Topics** | **Page No.** |
| 1 | [Problem Statement](#bookmark=id.1nmfz9jx6i5r) | 7 |
| 2 | [Objective](#bookmark=id.pss08oswrogb) | 7 |
| 3 | [Business Benefits](#bookmark=id.5l4wsi81spj7) | 7 |
| 4 | [Process Flow](#bookmark=id.4gi4kfmxolya) | 9 |
| 5 | [Component wise Architecture](#bookmark=id.1k5a0t7w6rn2) | 10 |
| 6 | [Potential data challenges & risks in doing the project](#bookmark=id.6hlsj5k1wun) | 19 |
| 7 | [Project Resource](#bookmark=id.3o0ei75j6fo1) | 20 |
| 8 | [Detailed Plan of Work](#bookmark=id.fvkp1gohfxkw) | 20 |
| 9 | [Code Structure](#bookmark=id.msvojlei7ai) | 21 |
| 10 | Screen Shots | 25 |
| 11 | [Interpretation](#bookmark=id.rrhbdtv3ud4) | 27 |
| 12 | [Conclusions / Recommendations](#bookmark=id.e2miin8zlz1) | 30 |
| 13 | [Future Work & Extension or Scope of improvements](#bookmark=id.lnvshpwvghzq) | 31 |
| 14 | [Bibliography / References](#bookmark=id.4hrxwme1tb05) | 32 |
| 15 | Appendix | 32 |
| 16 | Glossary | 33 |

**Problem Statement**

* Computer Vision systems deal with a high variety and volume of data, specifically images or videos. As a result, these systems need intricate techniques to make sense of the data and then make data driven decisions.
* The aim of this project is to create a model that can detect the basic events like swimming, jogging, soccer penally, basketball and boxing etc.
* Given a video, is it possible to recognize the action being performed in that video by an automatic system? An action recognition system is built on basic steps: first, the input video or sequence of frames; second, the extraction of low-level features from the frames; and finally, mid-level pose/gesture or action descriptions from low-level features.
* The model will be given a set of offline videos where in each video, a group of persons will be performing an action. The label of a video will be the action that is being performed in that video. The model will have to learn this relationship, and then it should be able to predict the label of an input (video) that it has never seen. Technically, the model would have to learn to differentiate between various individual events.

**Objective**

* In recent years, automatic human activity recognition and event detection have drawn much attention in the field of video analysis technology due to the growing demands from many applications, such as surveillance environments, entertainment environments and healthcare systems.
* Large amount of video in surveillance systems capable of processing video to automatically detect and recognize events. The main objective is the analysis of computer vision techniques and algorithms to predict automatic detection of specific events in video sequences

**Business Benefits**

**E-commerce**: Searching and advertising is one of the mostly used use case of image recognition in E-commerce industry. Image recognition powered by deep learning can provide us advance capabilities like personalized searches, customer analytics, social media and conversation commerce, etc. With the date they got from image recognition, businesses can find insights for campaigns and marketing strategies. It is also capable to find user’s sentiments and expressions. This data will help marketers getting more return on their investments.

**Social Media**: Social media is also getting enormous benefits from image recognition. Facebook is one of the best examples of it. Facebook’s image recognition can recognize your family members and friends, identify their names and will give you suggestion to tag them while you upload picture with them. Many other social media platforms like LinkedIn & Twitter also use image recognition and object recognition features. Image recognition in social media make searches easy and effective.

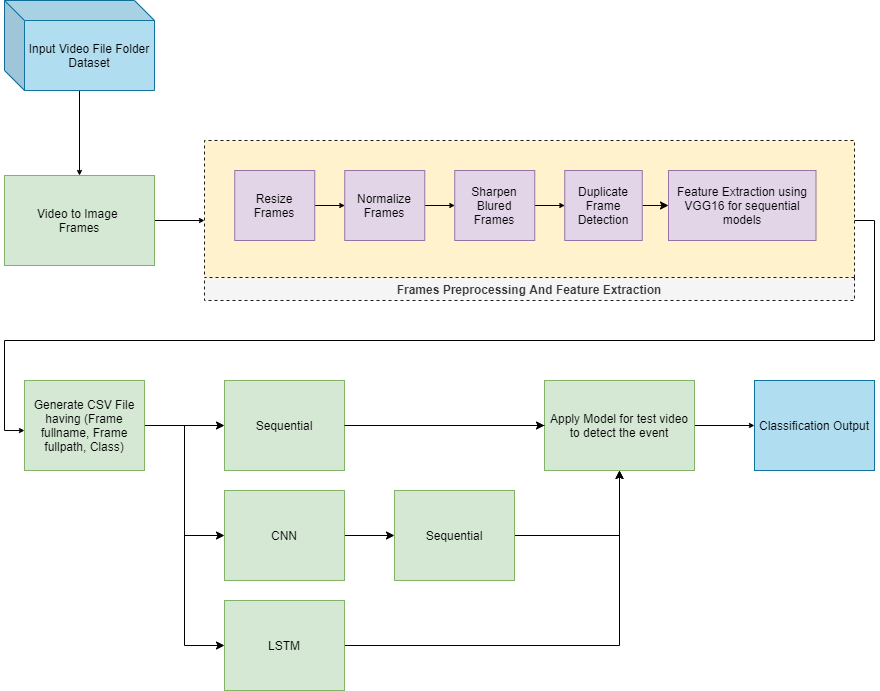
**Surveillance and Security**: In surveillance and security industry, image recognition has many applications. Many big companies and security department use facial recognition to ensure security and identify crime. This process includes scanning of almost million images and adding them into deep neural networks. Then the system analyzes the images and compare them to suspects. In many cases it has proven to be very successful & effective and helped solving crimes.

**Healthcare monitoring applications**: Basically, healthcare monitoring systems are designed based on the combination of one or more AR components such as fall detection, human tracking, security alarm and cognitive assistance components. Most of the healthcare systems use body-worn and contextual sensors that are placed on patients’ bodies and in their environment. Once help is needed, the system notifies the relevant parties (i.e. medical personnel) about the situation to assist the patient quickly. The E-safe fall detection and notification system[27](https://journals.sagepub.com/doi/10.1177/1550147716665520) has used the ZigBee-based wearable sensor system to automatically detect fall situations and notify the in-house correspondents via ZigBee technology.

**Gaming**: Other than these industries, there are many sectors which are taking leverages of Image recognition and computer vision. Image recognition has several powerful applications that create a great deal. What do you think about this technology and what are the use case which you would like to cover with image recognition?  Tell us your viewpoint and let us help you out with your requirements. From identifying requirements to delivering best machine learning model, Kevitt can bring you all the benefits that come with these technologies within your business.

**Machine Learning Process Flow**

**Process Pipeline**

****

* Read video data from folder and extracting image frames from each video using CV2.
* Preprocess extracted frames to resize, normalize, sharpen blurred frame, remove duplicate frames.
* Save resulting image to folder and generate CSV with data including frame name, frame full path and class.
* Using three different models (Sequential, CNN & LSTM) and compare the results
* Use pre-trained VGG16 model without top layers for feature extraction for training sequential model.
* Out classification result for each video.

**Data Preprocessing**

In data pre-processing step we are extracting image frames from video. We are also cherry picking every fifth frame to reduce the number of frames we are using to process. We are also performing following pre-processing activities

**Resize frames:** Frames are resized to 64x64 size

**Normalize frames:** Normalize the resized frame by dividing it with 255 so that each pixel value then lies between 0 and 1

**Sharpen blurred frames:** Detect blurred image and sharpen using CV2 filter2D sharpen kernel

**Duplicate frame detection:** Call pHash method on each frame to create hash per image and compare hashes to detect duplicate images.

**Feature extraction:** In case of sequential models, we are using VGG16 model to extract features and passing them to custom sequential model.

**Component wise Architecture**

**VGG16 Model:**

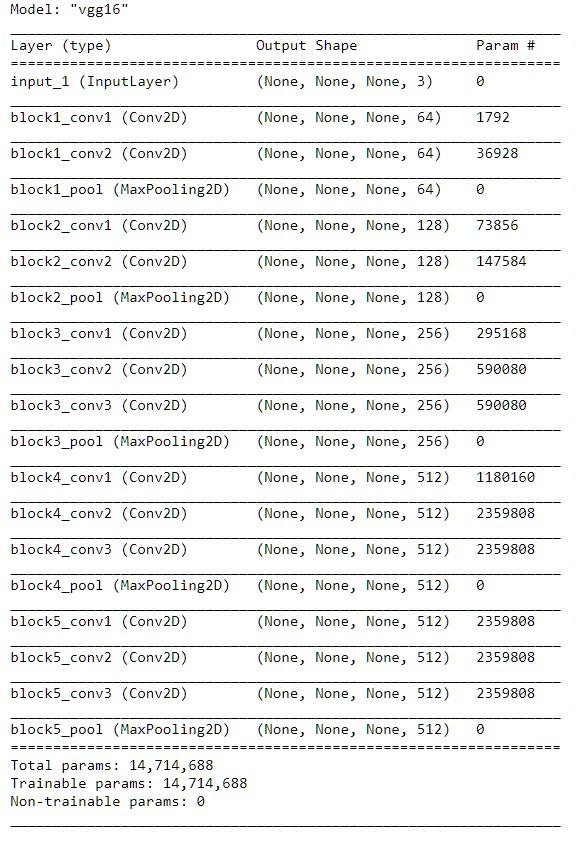
VGG16 is a convolutional network model developed and introduced by K. Simonyan and A. Zisserman from Oxford Visual Geometry Group, in the year 2014, through their article “Very Deep Convolutional Networks for Large-Scale Image Recognition.”. The VGG16 model achieved 92.7% accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes.

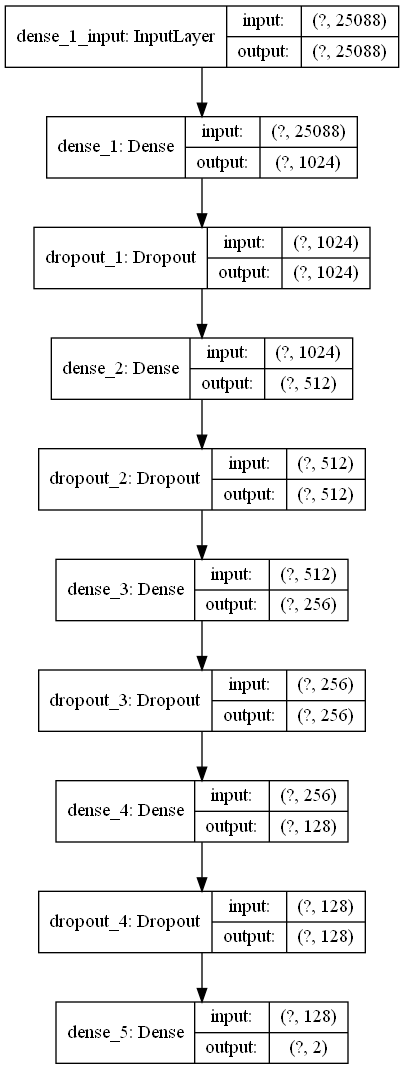
Table, timeline

Description automatically generated

The model contains 16 layers with 5 sets of convolutional layers followed by max-pooling layers after each set and 3 fully connected layers. The input to VGG is set to an RGB image of size 224x224x3. The first two layers have 64 channels of 3x3 filter size and the same padding. This is then followed by a max-pooling layer of stride 2,2 and two more convolutional layers of 256 filter size 3,3. Another max-pooling layer of stride 2,2 is added next. After that there are 2 sets of 3 convolution layer and a max pool layer. Each has 512 filters of 3, 3 sizes with the same padding. This image is then passed to the stack of two convolution layers. There is a padding of 1-pixel (same padding) done after each convolution layer to prevent the spatial feature of the image. After the convolution and max-pooling layer stack, we get a 7x7x512 feature map. We flatten this output to make it a 1x 25088 feature vector. After this, there are 3 fully connected layers; the first layer takes input from the last feature vector and outputs a (1, 4096) vector, second layer also outputs a vector of size 1, 4096, but the third layer output 1000 channels for 1000 classes. We can set the 3rd connected layer to output an arbitrary number of classes pertaining to our use case. The last three layers are also called top layers of the model. VGG16 can also be used for just feature extraction and transfer learning. This can be achieved by excluding top layers. All hidden layers are equipped with rectification (ReLU) non-linearity, and the output layer uses SoftMax to denote the classification as a probability.

Below is the model summary that shows VGG16 layers implementation:





As we can see, we have 14,714,688 trainable parameters when predicting two classes.

Training time increases exponentially with the neural network architecture increasing/deepening. In general, it could take hours/days to train a 3–5 layers neural network with a large-scale dataset. Consequently, deploying VGG from scratch on a large-scale dataset is a tiresome and computationally expensive task due to the depth and number of fully connected layers/nodes in the models’ architecture. Another challenge is that building VGG from scratch requires considerably large memory space and bandwidth since the size of ImageNet trained VGG-16 weights is 528 MB. However, instead of building a VGG from scratch, we can perform transfer learning. Transfer learning is technique to utilize the knowledge of weights and features of previously trained models to solve problems in similar area. For example, we can use transfer learning to a binary image classifier model trained from a pre-trained VGG16 model. To do so we remove the top layers and set all the layers from VGG16 model as untrainable. This will lock the model to use existing training data, and knowledge of known weights. We can now use VGG16 model to extract the features from a given set of training data and pass it on to a different model.

**Sequential Model:**

The Sequential model API is a way of creating deep learning models where an instance of the Sequential class is created, and model layers are created and added to it. A Sequential model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor.

Dense is a standard deeply connected layer type that works for most use cases. In a dense layer, all nodes in the previous layer connect to the nodes in the current layer. We specify number of neurons per dense layer, and an activation function. When an input shape is specified to the Dense function, an input layer is created and connected to a dense layer created with specified number of neurons.

Activation function constrains the output of a function(neuron) to values between 0 to 1 or -1 to 1. There are two types of activation functions – Linear activation function and Non-linear activation functions. Some examples of activation functions are ReLU, Sigmoid, Tanh, Leaky ReLU etc.

Dropout is a regularization method that approximates training many neural networks with different architectures in parallel. During training, some number of layer outputs are randomly ignored or “dropped out.” This has the effect of making the layer look-like and be treated-like a layer with a different number of nodes and connectivity to the prior layer. In effect, each update to a layer during training is performed with a different “view” of the configured layer. Usually, dropout is placed on the fully connected layers only because they are the one with the greater number of parameters and thus, they're likely to excessively co-adapting themselves causing overfitting. However, since it's a stochastic regularization technique, you can really place it everywhere

Table

Description automatically generated

**Diagram, table

Description automatically generated**

**CNN Model:**

A Convolutional Neural Network is an algorithm that takes an input image and assigns weights and biases to different aspects of the image to distinguish them from each other. The architecture of CNN can be compared to the neuron network in the human brain and was initially inspired by the arrangement of the visual cortex. The behavior of each neuron is defined by the weights assigned to each aspect of the input. [21]

When you input an image into a CNN, each layer generates several activation maps, and activation maps highlight the relevant features of the image input. Further, each neuron takes a patch of pixels as input, multiplies their color value by its weights, and sums them up, running them through the activation function.

Diagram

Description automatically generated

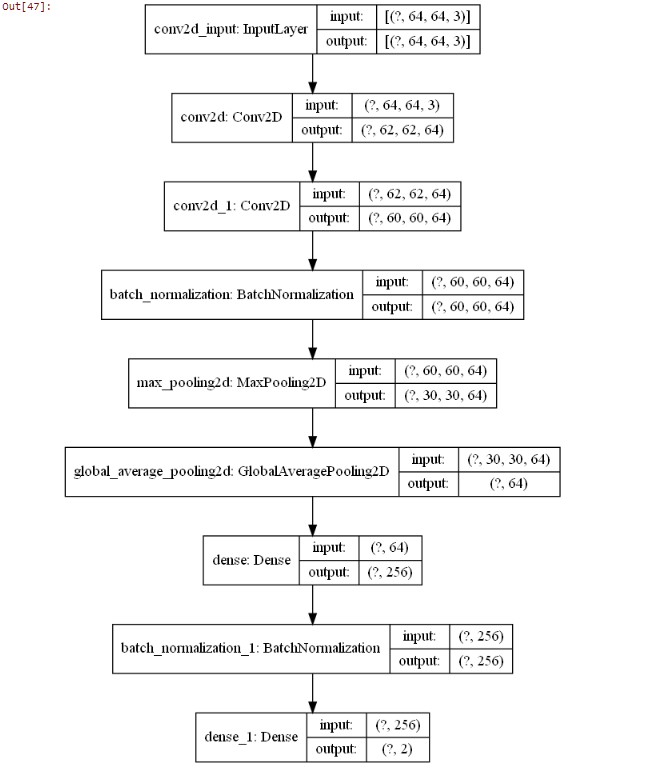
The above image shows what a convolution is. [2]

The first layer in the CNN usually detects basic features like horizontal, vertical edges, and diagonals of the images. As u move in deeper into the convolutional neural network, the layers start fetching objects like faces and more. This processed output is fed as input to the next layer, which further extracts more complex features. Based on the activation map of the final convolution layer, the classification layer outputs a set of confidence scores (values between 0 and 1)[2]

CNN have multiple layers, including convolutional layer, non-linearity layer, pooling layer, and fully connected layer. The convolutional and fully connected layers have parameters, but pooling and non-linearity layers don't have parameters. The CNN has an excellent performance in machine learning problems [3]

Table

Description automatically generated



**LSTM Model:**

Long Short Term Memory networks – usually just called LSTMs – are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter & Schmidhuber (1997), and were refined and popularized by many people in the following work. They work tremendously well on a large variety of problems and are now widely used.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

All recurrent neural networks have the form of a chain of repeating modules of neural networks. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.

Diagram, schematic

Description automatically generated

Graphical user interface, application

Description automatically generated

LSTM was created as the solution to short-term memory. They have internal mechanisms called gates that can regulate the flow of information. These gates can learn which data in a sequence is important to keep or throw away. By doing that, it can pass relevant information down the long chain of series to make predictions.

The core concept of LSTM is the cell state and its various gates. The cell state act as a transport highway that carries relevant information down the sequence chain. As the cell state goes on its journey, information gets added or removed via gates. The gates are different neural networks that decide which information is allowed on the cell state, and thus, the gates can learn which data is to be saved and which information to be discarded during training.

the tanh activation function is used squishes the value to be between -1 and 1. When vectors flow through the network, they undergo many transformations due to math operations.

Table

Description automatically generated

Diagram, table

Description automatically generated

**Potential data challenges & risks in doing the project**

* A first challenge associated with the complexity of real world dynamics that are demonstrated in such video streams, including changes in view point, illumination and quality.
* Although annotated image datasets are accepted, a smaller number of labeled datasets are available for video analytics.
* Lastly, massive, high-dimensional video stream analysis requires significantly higher computational resources than still imagery.
* Sports videos, in general, are acquired from different points, and the director decides to select a single stream for broadcasting. As a result, the broadcasted video stream is characterized by varying acquisition conditions like zooming in near the goalpost during a goal and zooming out to cover the entire field
* Despite the various difficulties linked with video analytics, the human brain can extract meaning and provide contextual learning in a limited time and from a limited set of training examples. From a computational viewpoint, event detection in a video sequence amounts to two fundamental steps,

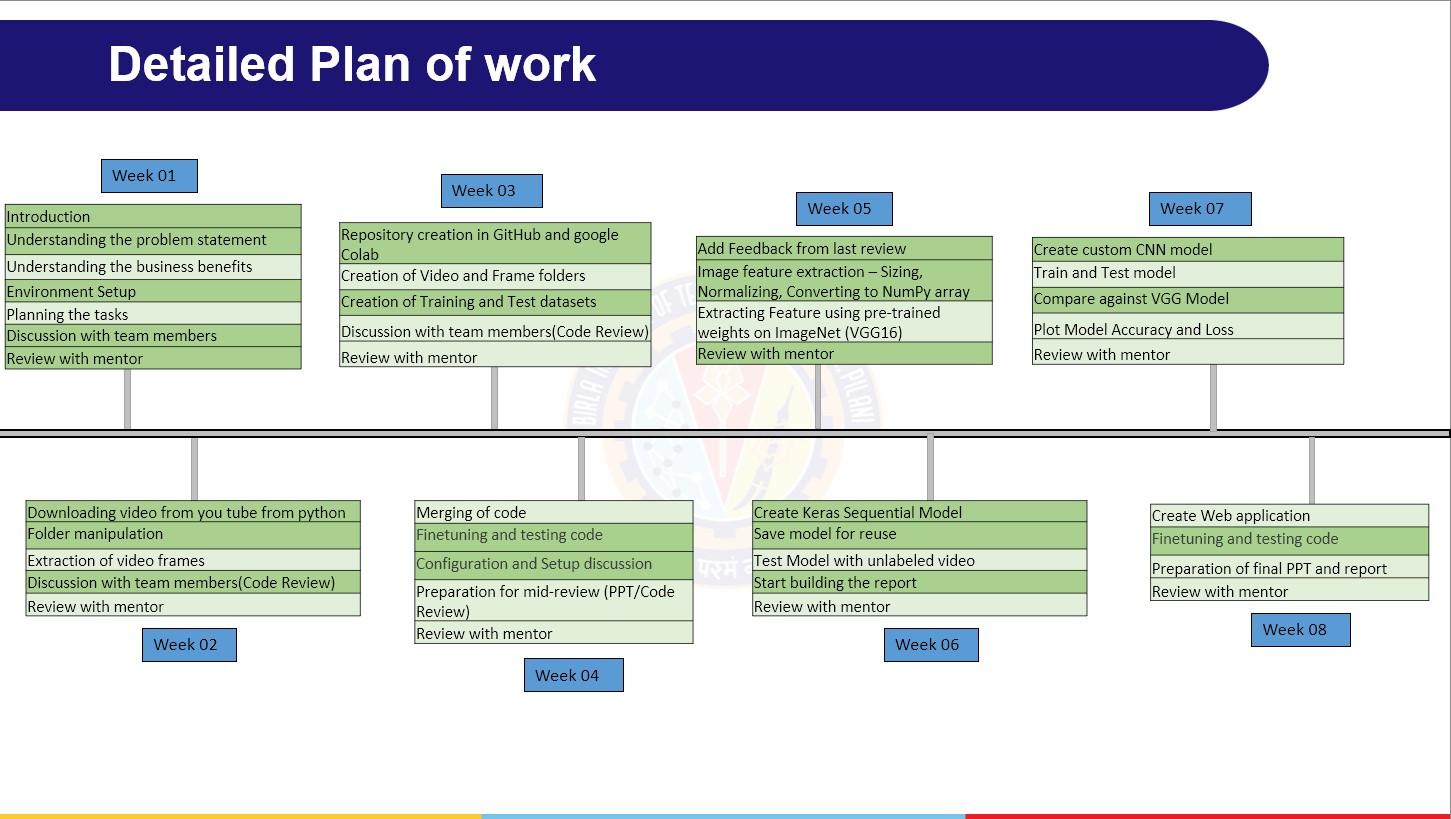
1. Spatio-temporal feature extraction
2. example classification.

* Since in advertisement video annotation we are mainly dealing with video files, so video should have the following characteristics for optimal video processing
  1. The video files should be in valid extensions like .mp4 or .avi files.
  2. The video should be in sufficient pace.
  3. Need large quality dataset to create and train models, on a particular event.
  4. Further improve the image quality in pre-processing steps using techniques like denoising, segmentation, morphology etc.
  5. Monotonous video data can overfit the model resulting poor actual event data accuracy.
  6. Duplicate data removal may skew the result of event detection.

**Project Resources**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SNo** | **Phase** | **Resource Type** | **Resource** | **Versions if Applicable** |
| 1 | All | People | Mentor, Faculty, TA, Student |  |
| 2 | All | Hardware / Laptop | 16GB RAM, Windows 10, Intel Core i7 |  |
| 3 | All | Software | Jupyter Notebook  Anaconda  Python  CV2 | 6.1.4  4.8.3  3.8.5  4.3.0 |
| 4 | Collaboration | Software | Google Meet  GitHub repository  WhatsApp  Google Colab | https://github.com/eiensatya7/cap4\_group7\_va  https://colab.research.google.com/notebooks |

**Detailed Plan of Work**

****

**Code** **Structure and Screenshots**

***Installation Steps:***

Please refer to “readme.txt”: to define the required package and version used in the project.

***Structure:***

***Timeline

Description automatically generated with medium confidence***

**Folder Structure:**

**.**

1) For running step 0 , step 1 and step 2

a. C:\1-GG\CAP4\EventDetection\1-ExampleSetting is the root folder which needs to be created

b. Paste Videos for actions from the UCF dataset in their respective folders with folder name as the required action . For Ex- Place all videos depicting basketball in a folder named basketball

c. Create a folder named “Client file” to keep the updated test\_BSvdo\_list.xlsx file

d. For step 2 a video file to test the model needs to be provided , paste the same in the root folder

2) For running step 3

**Folder Structure:**

a. C:\1-GG\CAP4\EventDetection\Dataset is the root folder which needs to be created

b. Create the following folders under root with the following names

i. CNN\_Apply\_vdo

1. Apply\_vdo (Will contain the video which can be used for testing)

2. OutPut\_vdo (script will create the video with tagged actions to test the model)

ii. SavedModel (The created model will be stored here)

c. VDO

i. Paste Videos for actions from the UCF dataset in their respective folders with folder name as the required action. For Ex- Place all videos depicting basketball in a folder named basketball

3) For running step 4

a. C:\1-GG\CAP4\EventDetection\Dataset-LSTM is the root folder which needs to be created

b. Create the following folders under root with the following names

i. CNNLSTM\_Apply\_vdo

1. Apply\_vdo (Will contain the video which can be used for testing)

2. OutPut\_vdo (script will create the video with tagged actions to test the model)

ii. SavedModel (The created model will be stored here)

c. VDO

i. Paste Videos for actions from the UCF dataset in their respective folders with folder name as the required action. For Ex- Place all videos depicting basketball in a folder named basketball

**Code Structure:**

.

step 0:

Filename: PCAMZC321\_Group5\_VIDEOANALYTICS\_PreProcessing\_Step-0. ipynb

Downloading and storing video from You Tube

creating folder and subfolder

video to frame conversion

blur and duplicate detection

pre-processing and storing frames (although only being used in first model)

step 1:

VGG Model

File Name: - PCAMZC321\_Group5\_VIDEOANALYTICS\_VGG16SequentialModelTraining\_Step-1.iypnb

creating the base model of pre-trained VGG16 model

Keras Neural Network Sequential Model

Training the video classification model

Evaluate trained model on the feature’s and label’s test sets

step 2:

File Name: - PCAMZC321\_Group5\_VIDEOANALYTICS\_VGG16SequentialModelOutputVideoCreation\_Step-2.iypnb

Applying model on VDO having no class

Constructing The Output YouTube Video Path

Event Prediction

Save prediction

Predict on Live Videos

step 3:

CNN model

File Name: -PCAMZC321\_Group5\_VIDEOANALYTICS\_CNNModelandVideoCreation\_Step-3. iypnb

Extract, Resize and Normalize Frames

Dataset Creation

Constructing the Model

Compile and Train the Model

Evaluating Trained Model

create Video

step 4:

LSTM model

File Name: - PCAMZC321\_Group5\_VIDEOANALYTICS\_LSTMModelandVideoCreation\_Step-4. iypnb

Creating frames from videos

creating model

load the model from disk

model\_evaluation\_loss, model\_evaluation\_accuracy

**Output Snapshot**

##### Graphical user interface, website Description automatically generated

**2.** A picture containing text, grass, athletic game, crowd

Description automatically generated

**3.** A picture containing text, grass, outdoor

Description automatically generated

**4. A picture containing text, grass, blue, screenshot

Description automatically generated**

**5 .A group of people playing basketball

Description automatically generated with medium confidence**

**Interpretation**

**Comparing Models**

|  | **FrameFilename** | **FullPathName** | **class** |
| --- | --- | --- | --- |
| **0** | v\_Basketball\_g01\_c01.avi | C:/1-GG/CAP4/EventDetection/1-ExampleSetting\V... | Basketball |
| **1** | v\_Basketball\_g01\_c01.avi | C:/1-GG/CAP4/EventDetection/1-ExampleSetting\V... | Basketball |
| **2** | v\_Basketball\_g01\_c01.avi | C:/1-GG/CAP4/EventDetection/1-ExampleSetting\V... | Basketball |
| **3** | v\_Basketball\_g01\_c01.avi | C:/1-GG/CAP4/EventDetection/1-ExampleSetting\V... | Basketball |
| **4** | v\_Basketball\_g01\_c01.avi | C:/1-GG/CAP4/EventDetection/1-ExampleSetting\V... | Basketball |
| **5** | v\_SoccerPenalty\_g01\_c01.avi | C:/1-GG/CAP4/EventDetection/1-ExampleSetting\V... | SoccerPenality |
| **6** | v\_SoccerPenalty\_g01\_c01.avi | C:/1-GG/CAP4/EventDetection/1-ExampleSetting\V... | SoccerPenality |
| **7** | v\_SoccerPenalty\_g01\_c01.avi | C:/1-GG/CAP4/EventDetection/1-ExampleSetting\V... | SoccerPenality |
| **8** | v\_SoccerPenalty\_g01\_c01.avi | C:/1-GG/CAP4/EventDetection/1-ExampleSetting\V... | SoccerPenality |

**Comparing Models**

Accuracy charts for CNN , VGG and LSTM models

1. CNN model accuracy chart 2. VGG model accuracy chart

Chart, line chart

Description automatically generated Chart, line chart

Description automatically generated

3. LSTM model accuracy chart

Chart, line chart

Description automatically generated

Loss charts for CNN, VGG, LSTM models

1.Loss chart for CNN Model 2.Loss chart for VGG model

Chart, line chart

Description automatically generated Chart, line chart

Description automatically generated

3. Loss chart for LSTM model

**Chart, line chart, histogram

Description automatically generated**

**CNN classification report**

**Table

Description automatically generated**

**LSTM classification report**

**Table

Description automatically generated**

**VGG classification report**

**Table

Description automatically generated**

**Conclusions / Recommendations**

In this project we have presented how machine learning models such as VGG16, Custom sequential models, CNN and LSTM model can be used to perform event detection in video data. We have seen video data is like image data except additional time dimension.

**Problem of overfitting:**

Overfitting occurs when you achieve a good fit of your model on the training data, while it does not generalize well on new, unseen data.

Overfitting can be identified by looking at validation metrics, like loss or accuracy. Usually, the validation metric stops improving after a certain number of epochs and begins to decrease afterwards. The training metric continues to improve because the model seeks to find the best fit for the training data.

The best option is to get more training data. Unfortunately, in real-world situations, you often do not have this possibility due to time, budget or technical constraints.

Another way to reduce overfitting is to lower the capacity of the model to memorize the training data. As such, the model will need to focus on the relevant patterns in the training data, which results in better generalization.

**Conclusions:**

In this project, deep learning-based transfer learning model, VGG-16, CNN, and LSTM approaches have been used for activity recognition in a video. Basic events like swimming, jogging, soccer penally, basketball and boxing are identified from the collected dataset through this pre-trained model. The highest accuracy achieved by the VGG model is 99% in activity recognition. Including spatial features with temporal features by incorporating the LSTM model as well, as the pre-trained VGG-16 model has just focused on spatial features on frame-level video activity recognition. While on the other hand, model accuracy fr CNN and LSTM are observed to be 88% and 93% respectively.

Based on the accuracy percentage, it can be concluded that VGG is the best model for event detection. VGG-16 is a pre-trained model and this trained data can be transferred to different data which would save the time and compute resources to learn a lot of features and our model will likely benefit from it. Further, LSTM is the next best model to predict activity recognition which can be confirmed from F1-score.

**Future Work & Extension or Scope of improvements**

1) Audio noise Reduction: use noise reduction technique from deep learning).

2) Logo Detection: (Rule based; Logo can be present besides company name).

3) Design frontend video upload functionality and store videos and images at some cloud and visualization of 360view to video. (with react and expose api for visualization).

4) Topic modelling.

5) Sentiment Analysis.

6) Object detection using GAN, CNN and encoder and decoder and other (using Generative Adversarial Network).

7) There is a huge scope of improvement in the approach used in order to address the problem. The data could be processed more efficiently. The pre-processing step should take care of

* the empty frames (with no human performing any action)
* Duplicate frames
* Noisy or blurred frame

Implementing the above steps could significantly improve the performance of the model by reducing the false positives rate.

The proposed model overfitted the training data after a certain number of epochs which suggests enough tuning to prevent overfitting, consequently improving the results.

8) Ability to transfer learning would yield better results by extracting features from the videos.

9) A simple web application can be developed which would capture any action as video and the model would then give real time predictions of the action being performed.

**Bibliography / References**

1. Albawi, S., Mohammed, T. and Al-Zawi, S., 2017. Understanding of a convolutional neural network. *2017 International Conference on Engineering and Technology (ICET)*,.
2. Rahman, L., Mohammed, N. and Al Azad, A., 2016. A new LSTM model by introducing biological cell state. *2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*,.
3. Yu, Y., Si, X., Hu, C. and Zhang, J., 2019. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Computation*, 31(7), pp.1235-1270.

4. Aljarrah, I. and Mohammad, D., 2018. Video content analysis using convolutional neural networks. *2018 9th International Conference on Information and Communication Systems (ICICS)*,.

5.Simonyan, K., 2021. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. [online] ResearchGate. Available at: <https://www.researchgate.net/publication/265385906\_Very\_Deep\_Convolutional\_Networks\_for\_Large-Scale\_Image\_Recognition> [Accessed 5 November 2021].

Other link:

1.Analytics Vidhya. 2021. *CNN for Deep Learning | Convolutional Neural Networks*. [online] Available at: <https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/> [Accessed 4 November 2021].

1. Dickson, B., 2021. *What are convolutional neural networks (CNN)?*. [online] TechTalks. Available at: <https://bdtechtalks.com/2020/01/06/convolutional-neural-networks-cnn-convnets/> [Accessed 4 November 2021].

**Appendix**

Jupyter Scripts (Refer code structure for complete breakdown):

* 1. PCAMZC321\_Group5\_VIDEOANALYTICS\_PreProcessing\_Step-0. Ipynb
  2. PCAMZC321\_Group5\_VIDEOANALYTICS\_VGG16SequentialModelTraining\_Step-1.iypnb
  3. PCAMZC321\_Group5\_VIDEOANALYTICS\_VGG16SequentialModelOutputVideoCreation\_Step-2.iypnb
  4. PCAMZC321\_Group5\_VIDEOANALYTICS\_CNNModelandVideoCreation\_Step-3. Iypnb
  5. PCAMZC321\_Group5\_VIDEOANALYTICS\_LSTMModelandVideoCreation\_Step-4. iypnb

**Glossary**

|  |  |
| --- | --- |
| VGG16 | Visual Geometry Group (Oxford) (VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford) |
| CNN | Convolution Neural Network (In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of artificial neural network, most commonly applied to analyze visual imagery) |
| LSTM | Long Short-Term Memory (Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning.) |
| ReLU | the rectifier or ReLU (Rectified Linear Unit) activation function is an activation function defined as the positive part of its argument: |

**Checklist of items for the Final report**

1. Is the Cover page in proper format? Y
2. Is the Title page in proper format? Y
3. Is the Certificate from the Mentor in proper format? Has it been signed? Y
4. Is Abstract included in the Report? Is it properly written? Y
5. Does the Table of Contents page include chapter page numbers? Y
6. Does the Report contain a summary of the literature survey? Y
   1. Are the Pages numbered properly? Y
   2. Are the Figures numbered properly? Y
   3. Are the Tables numbered properly? Y
   4. Are the Captions for the Figures and Tables proper? Y
   5. Are the Appendices numbered? Y
7. Does the Report have Conclusion / Recommendations of the work? Y
   * 1. Are References/Bibliography given in the Report? Y
     2. Have the References been cited in the Report? Y
     3. Is the citation of References / Bibliography in proper format? Y

**Note: A copy of this checklist should be included as the last page of the Final report. This checklist, duly completed and signed by the student, should also be verified and signed by the evaluators. Evaluators are requested to ensure that the students have submitted their reports properly.**