A Project Report

on

VIDEO STREAMING CAPTION GENERATOR USING DEEP LEARNING

Submitted in partial fulfillment of the requirements

for the award of degree of

BACHELOR OF TECHNOLOGY

in

Information Technology

by

N. Prasanna Laxmi (19WH1A1283)

B. Navya (19WH1A12A2)

T. Srilatha (20WH5A1207)

CH. Madhuri (20WH5A1208)

Under the esteemed guidance of

Mr. A. Rajashekar Reddy

Assistant Professor

Department of Information Technology

BVRIT HYDERABAD College of Engineering for Women

Rajiv Gandhi Nagar, Nizampet Road, Bachupally, Hyderabad – 500090

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad)

(NAAC 'A' Grade & NBA Accredited- ECE, EEE, CSE, IT)

June, 2023

DECLARATION

We hereby declare that the work presented in this project entitled “Video Streaming Caption Generator Using Deep Learning” submitted towards completion of the Project in IV year II sem of B.Tech IT at “BVRIT HYDERABAD College of Engineering for Women, Hyderabad” is an authentic record of our original work carried out under the esteemed guidance of Mr. A. Rajashekar Reddy, Assistant Professor, Department of Information Technology.

N. Prasanna Laxmi (19WH1A1283)

B. Navya (19WH1A12A2)

T. Srilatha (20WH5A1207)

CH. Madhuri (20WH5A1208)

BVRIT HYDERABAD

College of Engineering for Women

Rajiv Gandhi Nagar, Nizampet Road, Bachupally, Hyderabad – 500090

(Affiliated to Jawaharlal Nehru Technological University Hyderabad)

(NAAC 'A' Grade & NBA Accredited- ECE, EEE, CSE IT)

CERTIFICATE

This is to certify that the major-project report on **" Video Streaming Caption Generator Using Deep Learning "** is a bonafide work carried out by **N. Prasanna Laxmi (19WH1A12083), B. Navya (19WH1A1240), T. Srilatha (20WH5A1207)** and **CH. Madhuri (20WH5A1208)** in the partial fulfillment for the award of B.Tech degree in **Information Technology, BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad** affiliated to Jawaharlal Nehru Technological University, Hyderabad under my guidance and supervision.

The results embodied in the major project work have not been submitted to any other university or institute for the award of any degree or diploma.

Internal Guide	**Head of the Department**
Mr.A.Rajashekar Reddy	**Dr. Aruna Rao S L**
Assistant Professor	**Professor & HoD**
Department of IT	**Department of IT**

External Examiner

ACKNOWLEDGEMENT

We would like to express our profound gratitude and thanks to **Dr. K. V. N. Sunitha, Principal, BVRIT HYDERABAD** for providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. Aruna Rao S L, Professor & Head, Department of IT, BVRIT HYDERABAD** for all the timely support, constant guidance and valuable suggestions during the period of our project.

We are extremely thankful and indebted to our internal guide, **Mr. A. Rajashekar Reddy, Assistant Professor, Department of IT, BVRIT HYDERABAD** for his constant guidance, encouragement and moral support throughout the project.

Finally, we would also like to thank our Project Coordinators **Dr. P. Kayal, Associate Professor and Ms. K. S. Niraja, Assistant Professor**, all the faculty and staff of Department of IT who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

N. Prasanna Laxmi (19WH1A1283)

B. Navya (19WH1A12A2)

T. Srilatha (20WH5A1207)

CH. Madhuri (20WH5A1208)

ABSTRACT

The contents of an video are generated automatically which involves computer vision and NLP (Natural Language Processing). The neural model which is regenerative, is created. It depends on computer vision and machine translation. This model is used to generate natural sentences which eventually describes the video. This model consists of Convolutional Neural Network(CNN). The CNN is used for feature extraction from video and generate captions. The model is trained in such a way that if input video is given to model it generates captions which nearly describes the video. The accuracy of model and smoothness or command of language model learns from video descriptions is tested on different datasets. These experiments show that model is frequently giving accurate descriptions for an input video.

LIST OF FIGURES

CONTENTS

1. Introduction

The people communicate through language, whether written or spoken. They often use this language to describe the visual world around them. videos, signs are another way of communication and understanding for the physically challenged people. The generation of descriptions from the video automatically in proper sentences is a very difficult and challenging task, but it can help and have a great impact on visually impaired people for better understanding of the description of videos on the web. A good description of an video is often said for 'Visualizing a picture in the mind'. The creation of an video in mind can play a significant role in sentence generation. Also, human can describe the video after having a quick glance at it. The progress in achieving complex goals of human recognition will be done after studying existing natural video descriptions. This task of automatically generating captions and describing the video is significantly harder than video classification and object recognition. The description of an video must involve not only the objects in the video, but also relation between the objects with their attributes and activities shown in videos.

Most of the work done in visual recognition previously has concentrated to label videos with already fixed classes or categories leading to the large progress in this field. Eventually, vocabularies of visual concepts which are closed, makes a suitable and simple model for assumption. These concepts appear widely limited after comparing them with the tremendous amount of thinking power which human possesses. However, the natural language like English should be used to express above semantic knowledge, that is, for visual understanding language model is necessary. The relation between visual importance and descriptions moves to the text summarization problem in natural language processing (NLP).

Machine Learning is a very vast subject and also subset of Artificial intelligence. Deep Learning can be described as subset of Machine learning, that is capable of learning from unstructured and unla- beled data. In recent times Deep learning has drastically changed the world of Computer Vision. By using the features of deep learn- ing features and representations, machines can give comparable or better performance than human beings in object recognition,image classification and video segmentation, but still there are de- velopments needed in segments like image and video captioning.

Video captioning becomes very difficult as there are complex scenes in videos and diverse kinds of objects are present around which sometimes causes problem for captioning.

In several recent applications, deep learning has transformed computer vision. A machine can match

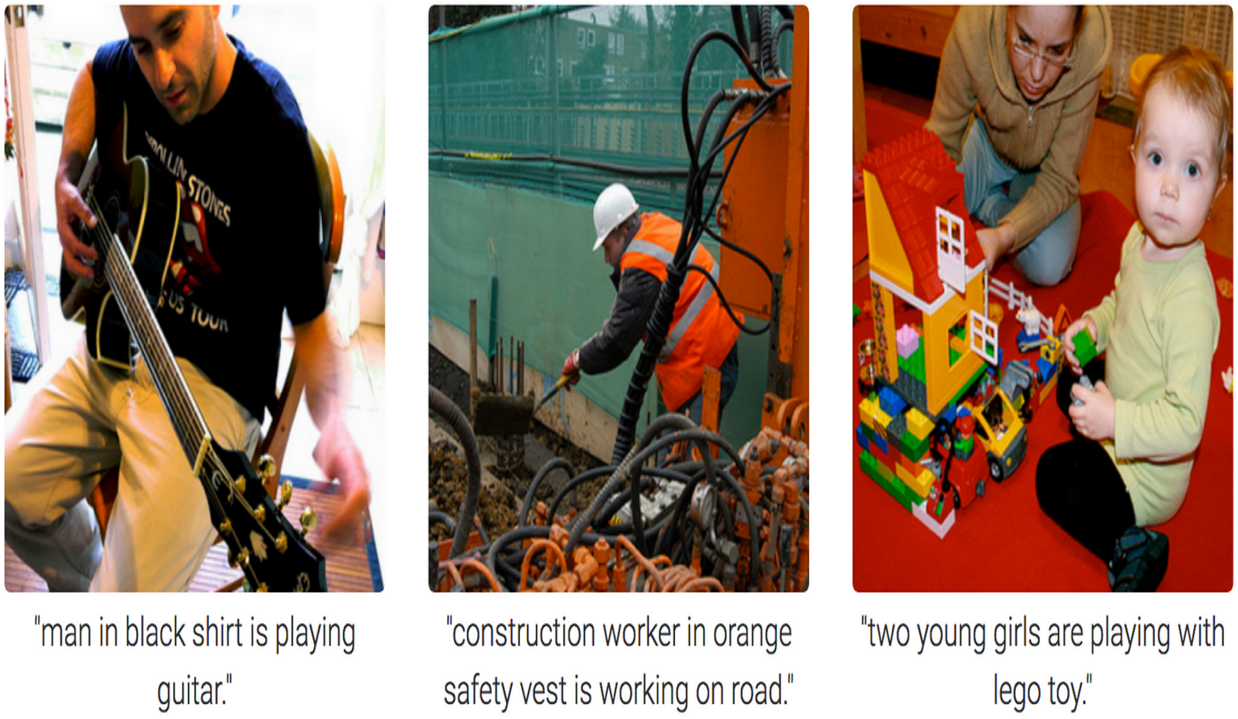

"man in black shirt is playing guitar."
"construction worker in orange safety vest is working on road."
"two young girls are playing with lego toy."

Figure 1.1: Video Captioning

or even surpass human performance in a variety of tasks, including picture classification, object identification, and video segmentation, by learning deep features and representations. But in the past twoyears, literature has paid a great deal of attention to activities like captioning images and videos, which continue to be difficult. Captioning is extremely challenging due to the nature of a video stream's high temporal dependencies, several scenes in a complicated video, and a variety of objects and events. Nevertheless, a number of systems and techniques have been put forth that significantly advance research in video description. Building on prior achievements, this thesis work creates strong captioning frameworks that can provide captions for both straightforward and difficult videos automatically. The act of captioning videos has become more and more common in recent years. Short form video has solidified its position as an essential component of our daily lives with the rise of video sharing websites like YouTube, Twitch, and Instagram Reels. In fact, more than 500 million people use Facebook every day to watch videos, according to Forbes! 72 hours' worth of fresh video being uploaded to YouTube every minute. Artificial intelligence (AI) video solutions are now necessary due to the popularity boom of videos.

Visual perception and language expression are two key capabilities of human intelligence, and video captioning is a perfect example towards learning from human to bridge vision and language. The goal of video captioning is to automatically describe the visual content of a video with natural language. Practical applications of automatic caption generation include leveraging descriptions for video indexing or retrieval, and helping those with visual impairments by transforming visual signals into information that can be communicated via text-to-speech technology.

1.1 Objective

Video Captioning is a task of automatic captioning a video by understanding the action and event in the video which can help in the retrieval of the video efficiently through text.For humans, describing a scene in an image or video clip is a very difficult task. Due to the additional difficulty of identifying the objects and actions in the image and constructing a brief meaningful sentence based on the contents found, video captioning requires more effort than image recognition. The development of this method creates a wide range of real-world opportunities, including the assistance of those with various degrees of visual impairment, self-driving cars, automatic video subtitling, video surveillance, and more.

1.2 Problem Definition

The main aim of the project is used to generate captions for the video stream. The video stream is given as input, the features of the images will be extracted by using CNN and captions will be generated.

2. Literature Survey

Object-Aware Aggregation With Bidirectional Temporal Graph for Video Captioning

Junchao Zhang ,Yuxin Peng proposed that Video captioning aims to automatically generate natural language descriptions of video content, which has drawn a lot of attention recent years. Generating accurate and fine-grained captions needs to not only understand the global content of video, but also capture the detailed object information. Meanwhile, video representations have great impact on the quality of generated captions. Thus, it is important for video captioning to capture salient objects with their detailed temporal dynamics, and represent them using discriminative spatio-temporal representations. In this paper, we propose a new video captioning approach based on object-aware aggregation with bidirectional temporal graph (OA-BTG), which captures detailed temporal dynamics for salient objects in video, and learns discriminative spatio-temporal representations by performing object-aware local feature aggregation on detected object regions. The main novelties and advantages are: (1) Bidirectional temporal graph: A bidirectional temporal graph is constructed along and reversely along the temporal order, which provides complementary ways to capture the temporal trajectories for each salient object. (2) Object-aware aggregation: Learnable VLAD (Vector of Locally Aggregated Descriptors) models are constructed on object temporal trajectories and global frame sequence, which performs object-aware aggregation to learn discriminative representations. A hierarchical attention mechanism is also developed to distinguish different contributions of multiple objects. Experiments on two widely-used datasets demonstrate our OA-BTG achieves state-of-the-art performance in terms of BLEU@4, METEOR and CIDEr metrics.

End-to-End Video Captioning With Multitask Reinforcement Learning

Lijun Li,Boqing Gong proposed a multitask reinforcement learning approach to training a video captioning model in an E2E manner. Our main idea is to mine and construct as many effective tasks as possible from the human captioned videos such that they can jointly regulate the search space of the encoder-decoder network, from which an E2E video captioning model can be found and generalized to the testing ing phase. The auxiliary tasks consist of two broad types: to predict the attributes extracted from the captions of the training videos and to maximize the rewards defined from the reinforcement learning perspective. When the training set is relatively small for the big encoder-decoder network, it is important to mine as much supervision as possible from the limited data so that it helps reduce the search space for the main task of interest.Although end-to-end (E2E) learning has led to impressive progress on a variety of visual understanding tasks, it is often impeded by hardware constraints (e.g., GPU memory) and is prone to overfitting. When it comes to video captioning, one of the most challenging benchmark tasks in computer vision, those limitations of E2E learning are especially amplified by the fact that both the input videos and output captions are lengthy sequences. Indeed, state-of-the-art methods for video captioning process video frames by convolutional neural networks and generate captions by unrolling recurrent neural networks. If we connect them in an E2E manner, the resulting model is both memory-consuming and data-hungry, making it extremely hard to train. In this paper, we propose a multitask reinforcement learning approach to training an E2E video captioning model. The main idea is to mine and construct as many effective tasks (e.g., attributes, rewards, and the captions) as possible from the human captioned videos such that they can jointly regulate the search space of the E2E neural network, from which an E2E video captioning model can be found and generalized to the testing phase. To the best of our knowledge, this is the first video captioning model that is trained end-to-end from the raw video input to the caption output. Experimental results show that such a model outperforms existing ones to a large margin on two benchmark video captioning datasets.

Streamlined Dense Video Captioning

Jonghwan Mun,Linjie Yang,Zhou Ren,Ning Xu proposed a novel framework of detecting event sequences for dense video captioning. The proposed event sequence generation network allows the captioning network to model temporal dependency between events and generate a set of coherent captions to describe an episode in a video.Dense video captioning is an extremely challenging task since accurate and coherent description of events in a video requires holistic understanding of video contents as well as contextual reasoning of individual events. Most existing approaches handle this problem by first detecting event proposals from a video and then captioning on a subset of the proposals. As a result, the generated sentences are prone to be redundant or inconsistent since they fail to consider temporal dependency between events. To tackle this challenge, we propose a novel dense video captioning framework, which models temporal dependency across events in a video explicitly and leverages visual and linguistic context from prior events for coherent storytelling. This objective is achieved by 1) integrating an event sequence generation network to select a sequence of event proposals adaptively, and 2) feeding the sequence of event proposals to our sequential video captioning network, which is trained by reinforcement learning with two-level rewards - at both event and episode levels - for better context modeling. The proposed technique achieves outstanding performances on ActivityNet Captions dataset in most metrics.

A Fast Feature Extraction Algorithm for Image and Video Processing

Sadiq H.Abdulhussain,And Rahman Ramli proposed his a new method of applying the transformation function on the partitioned blocks to extract features from an image or video frame directly without using the processes of the traditional method.Medical images and videos are utilized to discover, diagnose and treat diseases. Managing, storing, and retrieving stored images effectively are considered important topics. The rapid growth of multimedia data, including medical images and videos, has caused a swift rise in data transmission volume and repository size. Multimedia data contains useful information; however, it consumes an enormous storage space. Therefore, high processing time for that sheer volume of data will be required. Image and video applications demand for reduction in computational cost (processing time) when extracting features. This paper introduces a novel method to compute transform coefficients (features) from images or video frames. These features are used to represent the local visual content of images and video frames. We compared the proposed method with the traditional approach of feature extraction using a standard image technique. Furthermore, the proposed method is employed for shot boundary detection (SBD) applications to detect transitions in video frames. The standard TRECVID 2005, 2006, and 2007 video datasets are used to evaluate the performance of the SBD applications. The achieved results show that the proposed algorithm significantly reduces the computational cost in comparison to the traditional method.

Image Captioning: A Comprehensive Survey

Himanshu Sharma, Manmohan Agrahari, Sujeet Kumar Singh, Mohd Firoj and Ravi Kumar Mishra proposed that the primary purpose of image captioning is to generate a caption for an image. Image captioning needs to identify objects in image, actions, their relationship and some silent feature that may be missing in the image. After identification the next step is to generate a most relevant and brief description for the image that must be syntactically and semantically correct. It uses both computer vision concepts for identification of objects and natural language processing methods for description. It's difficult for a machine to imitate human brain ability however researches in this field have shown a great achievement. Deep learning techniques are enough capable to handle such problems using CNN and LSTM. It can be used in many intelligent control systems and IOT based devices. In this survey paper, we are presenting different approaches of image captioning such as retrieval based, template based and deep learning based as well as different evaluation techniques. CNN-RNN framework based image captioning technique have two drawbacks in training phase. First drawback is each caption gets equal importance without their individual importance and second drawback is during caption generation objects may not be correctly recognized. EncoderDecoder framework, in this paper we have also suggested called Reference based long short term memory (R-LSTM) that main aim is by implementing reference information to give more descriptive caption for a query image. According to relation between image and words during the training phase, different weights are assigned. In addition to maximizing the agreement-score among the captions produced through the captioning methods and the reference data from the adjoining images of the intentional images that can limit the e issue of not recognize correctly an image.

Recurrent convolutional video captioning with global and local attention

Tao Jin,Yingming Li,Zhongfei Zhang proposed that Video captioning with encoder–decoder structures has been extensively studied in the recent literature, where a great deal of work focuses on multimodal features and attention mechanisms. Most of the previous work uses only the global temporal features, such as image, motion, and audio features, and ignores the local semantic features existing extensively in the video data. Furthermore, it is difficult to fully utilize the local features due to frame-to-frame redundancy. In this paper, we propose to combine global temporal features and local object-based features in a complementary way to develop a multimodal attention mechanism (global–local attention mechanism). Based on this attention mechanism, we introduce a novel video captioning method, Recurrent Convolutional Video Captioning with Global and Local Attention (RCGL). Further, both LSTM and 1D CNN are incorporated into the decoder to improve the long-range dependency. The experimental results on two standard datasets, MSVD and MSR-VTT, demonstrate that RCGL outperforms the state-of-the-art in four common metrics.

Video Description: A Survey of Methods, Datasets, and Evaluation Metrics

Nayyer Aafaq, Ajmal Saeed Mian, Wei Liu proposed that the Video description is the automatic generation of natural language sentences that describe the contents of a given video. It has applications in human-robot interaction, helping the visually impaired and video subtitling. The past few years have seen a surge of research in this area due to the unprecedented success of deep learning in computer vision and natural language processing. Numerous methods, datasets, and evaluation metrics have been proposed in the literature, calling the need for a comprehensive survey to focus research efforts in this flourishing new direction. This article fills the gap by surveying the state-of-the-art approaches with a focus on deep learning models; comparing benchmark datasets in terms of their domains, number of classes, and repository size; and identifying the pros and cons of various evaluation metrics, such as SPICE, CIDEr, ROUGE, BLEU, METEOR, and WMD. Classical video description approaches combined subject, object, and verb detection with template-based language models to generate sentences. However, the release of large datasets revealed that these methods cannot cope with the diversity in unconstrained open domain videos. Classical approaches were followed by a very short era of statistical methods that were soon replaced with deep learning, the current state-of-the-art in video description. Our survey shows that despite the fast-paced developments, video description research is still in its infancy due to the following reasons: Analysis of video description models is challenging, because it is difficult to ascertain the contributions towards accuracy or errors of the visual features and the adopted language model in the final description. Existing datasets neither contain adequate visual diversity nor complexity of linguistic structures. Finally, current evaluation metrics fall short of measuring the agreement between machine-generated descriptions with that of humans. We conclude our survey by listing promising future research directions.

3. SYSTEM DESIGN

To implement the project Deep Learning Techniques was used for Video Streaming Cation Generator. After referring to the research papers the following are the finalized technologies to design the project. Below mentioned is the System Design of the project explained in detail.

3.1 Project Flow

Video is given as input through the CNN algorithm the features of the images which are present in the video will be extracted and then the suitable captions will be generated for the video.

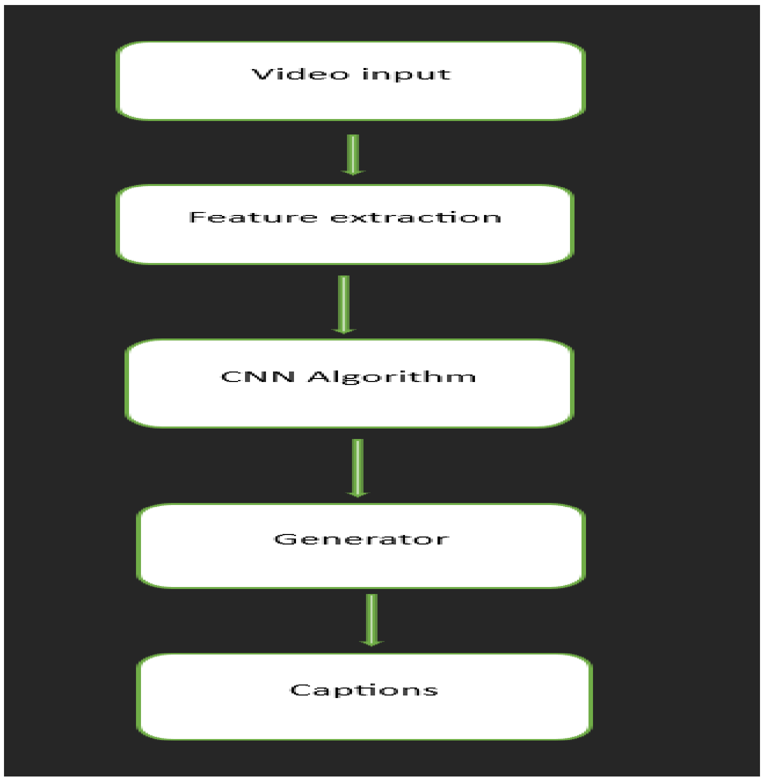


Figure 3.1.1: Project Flow

3.2 CNN Algorithm

A convolutional neural network (CNN/ConvNet) is a type of deep neural network that is often used to analyse visual imagery in deep learning. When we think about neural networks, we typically think of matrix multiplications, but this is not the case with ConvNet. It employs a technique known as Convolution. Convolution is a mathematical operation on two functions that yields a third function that explains how the shape of one is modified by the other. Convolutional neural networks are composed

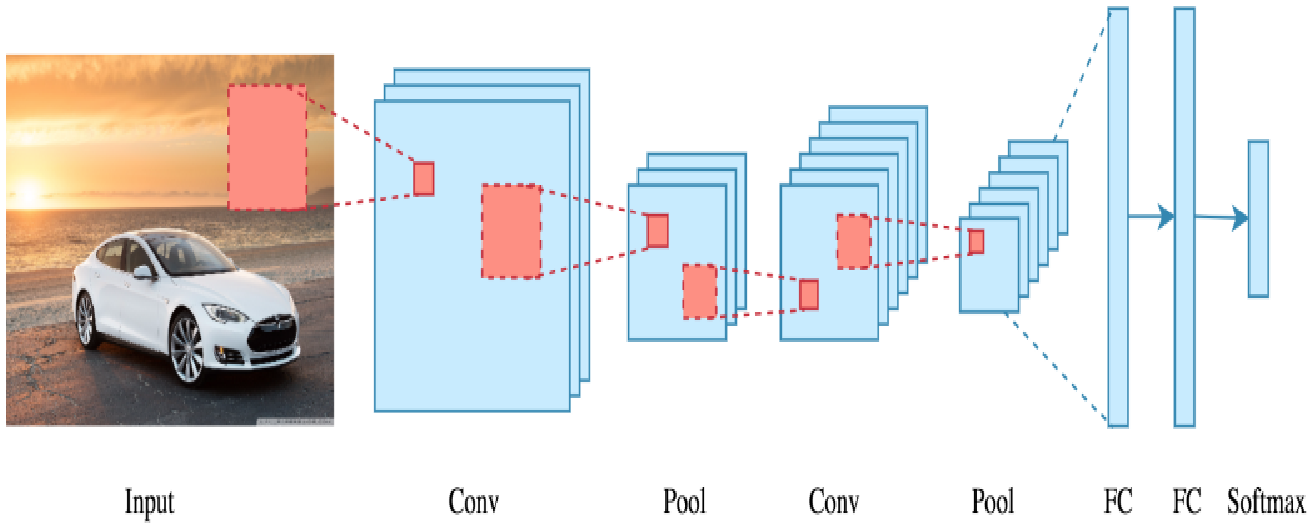


Figure 3.2.1: CNN Layers

of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value. When you input an image in a ConvNet, each layer generates several activation functions that are passed on to the next layer.

The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational

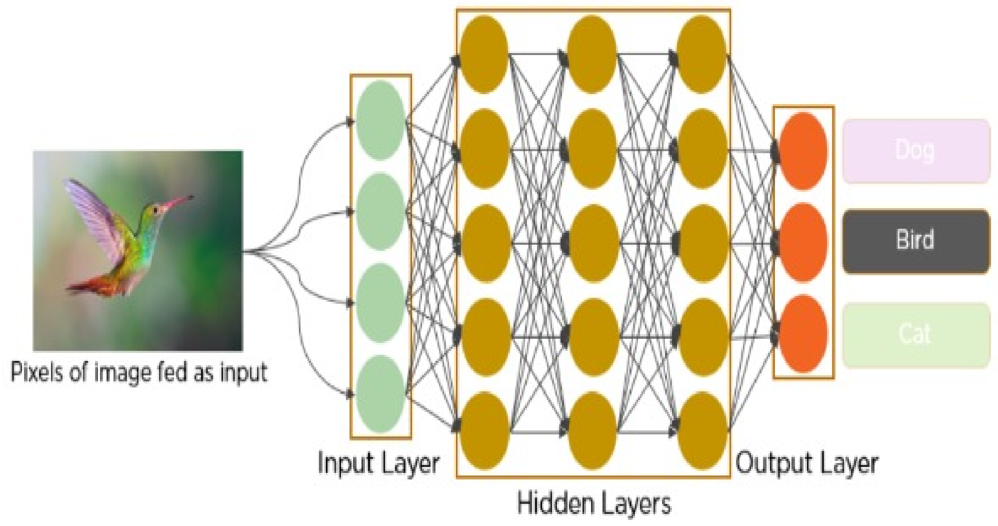


Figure 3.2.2: CNN Algorithm

edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc.When you feed an image into a ConvNet, each layer generates a set of activation functions that are then passed on to the next layer. Typically, the first layer extracts basic properties such as horizontal or diagonal edges. This output is forwarded to the following layer, which detects more complicated features such as corners or combinational edges. As we progress deeper into the network, it can recognise more complex aspects such as objects, faces, and so on. The classification layer generates a series of confidence scores (numbers between 0 and 1) based on the activation map of the final convolution layer, indicating how likely the image is to belong to a çlass.”For example, if a ConvNet detects cats, dogs, and horses, the final layer’s output is the likelihood that the input image contains any of those animals. The Pooling Layer, like the Convolutional Layer, is in charge of lowering the spatial size of the Convolved Feature. By lowering the size, the computer power required to process the data is reduced. Average pooling and maximum pooling are the two types of pooling. So far, I’ve only used Max Pooling and haven’t had any issues. Max Pooling finds the maximum value of a pixel from a region of the image covered by the kernel. Max Pooling is also a noise suppressant. It discards all noisy activations and conducts de-noising as well as dimensionality reductuction. Average Pooling, on the other hand, returns the average of all the values from the portion of the image covered by the Kernel. Average Pooling is just a noise suppression strategy that reduces dimensionality. As a result, we can conclude that Max Pooling outperforms Average Pooling.

1. Convolutional Layer

The most significant component of a Convolutional Neural Network (CNN) is the convolutional layer, which accounts for the majority of computational operations. This layer is important for extraction of both high-level and low-level features. High-level features include edges and picture input, whereas low-level features include colour, grade-orientation, and edges, among other things. To achieve full convolution, input is multiplied element by element to a large number of tiny sized sliding windows (kernels or filters) that are employed by each Convolutional Layer. As an example, the size of the input RGB CI-FAR-10 image is 32 x 32 x 3. With 16 filters and stride 1 of the very first layer, the size is 5 x 3 x 3, and the output is 28 x 28 x 16, in which the sample image is to be convolved. The image is padded with zeroes so that the output dimension matches the input dimension. There are three major parameters in the convolutional layer that affect the output volume: number of zero padding, depth, and stride. The stride controls the sliding of the filter across the input; the magnitude of the stride is inversely proportional to the spatial size of the output. The depth is determined by the number of filters used; certain new features are learned by the filter from the input. Zero padding is used to reduce the size of the input volume.

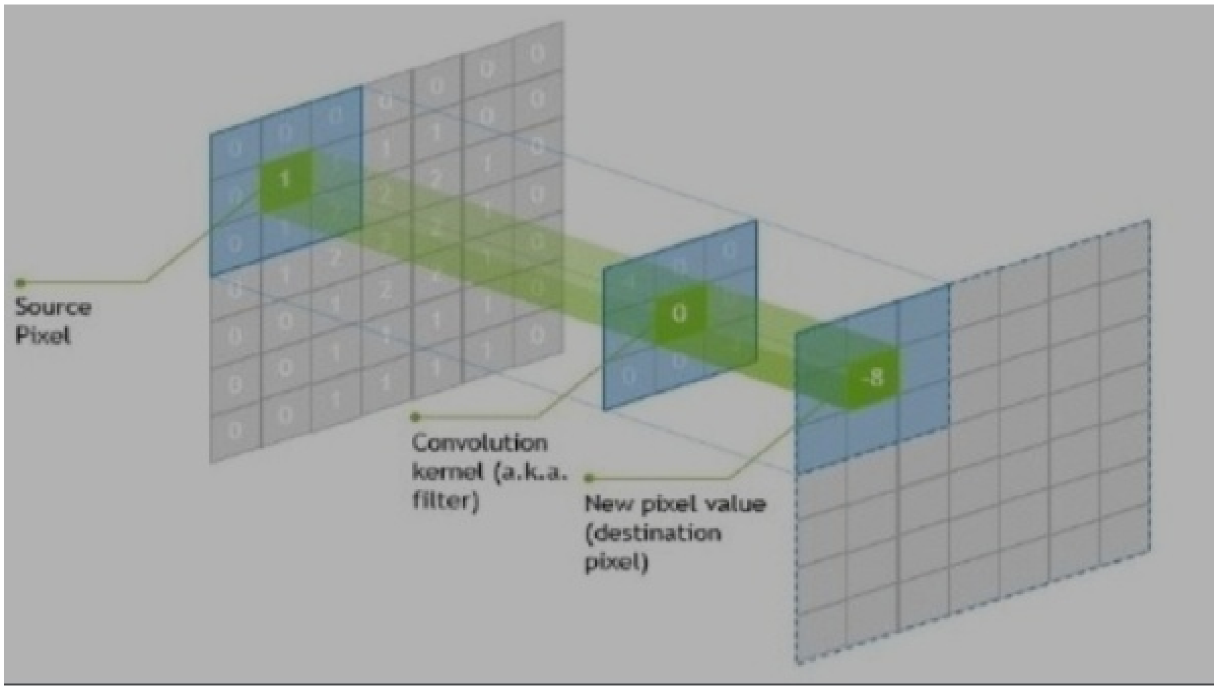

Source Pixel
Convolution kernel (a.k.a. filter)
New pixel value (destination pixel)

Figure 3.2.3: Convolutional layer

2. Pooling Layer

The pooling layer's job is to reduce the network's dimensionality. They are placed at regular intervals between each convolutional layer. This layer is also very important in keeping. Pooling layers are used to reduce the dimensions of the feature maps. Thus, it reduces the number of parameters to learn and the amount of computation performed in the network. The pooling layer summarises the features present in a region of the feature map generated by a convolution layer. So, further operations are performed on summarised features instead of precisely positioned features generated by the convolution layer. This makes the model more robust to variations in the position of the features in the input video image.

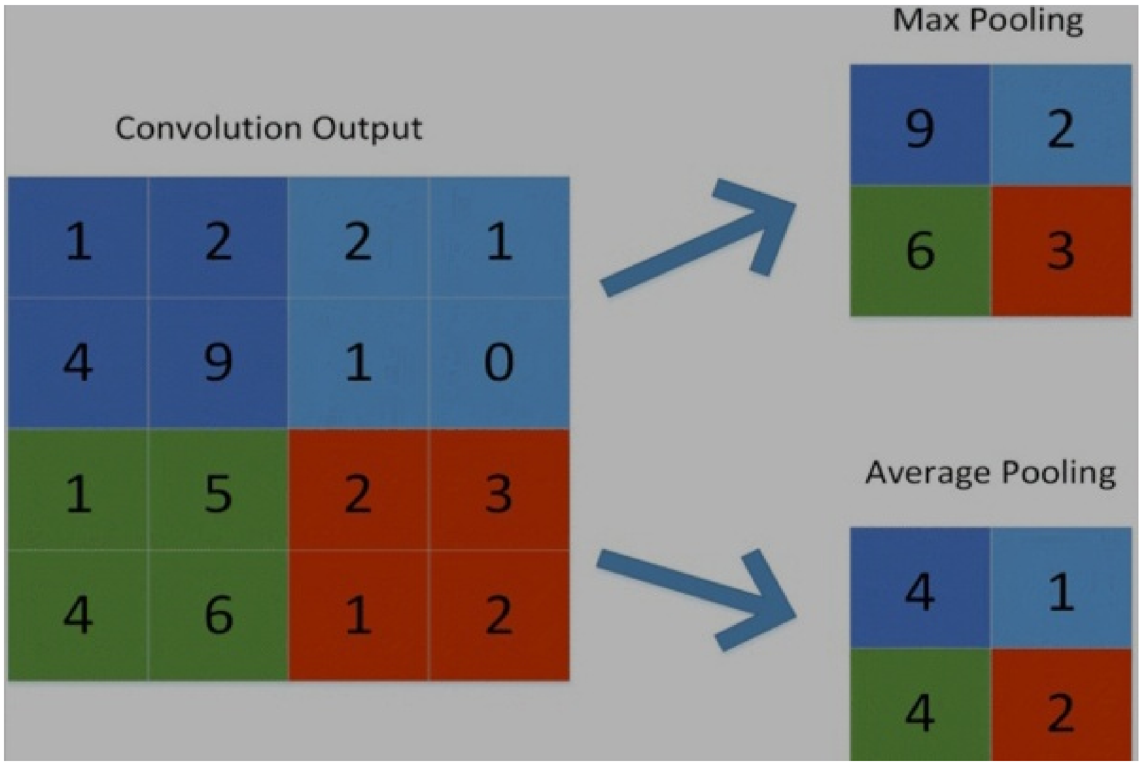


Figure 3.2.4: Pooling

3. ReLU function

Effective model training by utilising key properties such as rotational and positional invariance. Pooling Layers are classified into two types: Max-Pooling and Average Pooling. Max pooling returns the greatest value for the section of the image that is hidden by the kernel and also conducts dimension reduction. It is also known as sounds Suppressant because to its ability to suppress sounds. In terms of Average pooling, it conducts dimensionality reduction and also returns the average values from the picture that are covered by the Kernel. After studying both methods of pooling, it was discovered that the former, i.e., max-pooling, outperforms the latter, i.e., average-pooling. Convolutional Layer and Pooling Layer make up the k-th layer of a Convolutional Neural Network.The number of Convolutional Layers can be changed depending on the image's complexity. The number of Convolutional Layers can be raised to capture the image's low-level information, but takes more processing power.

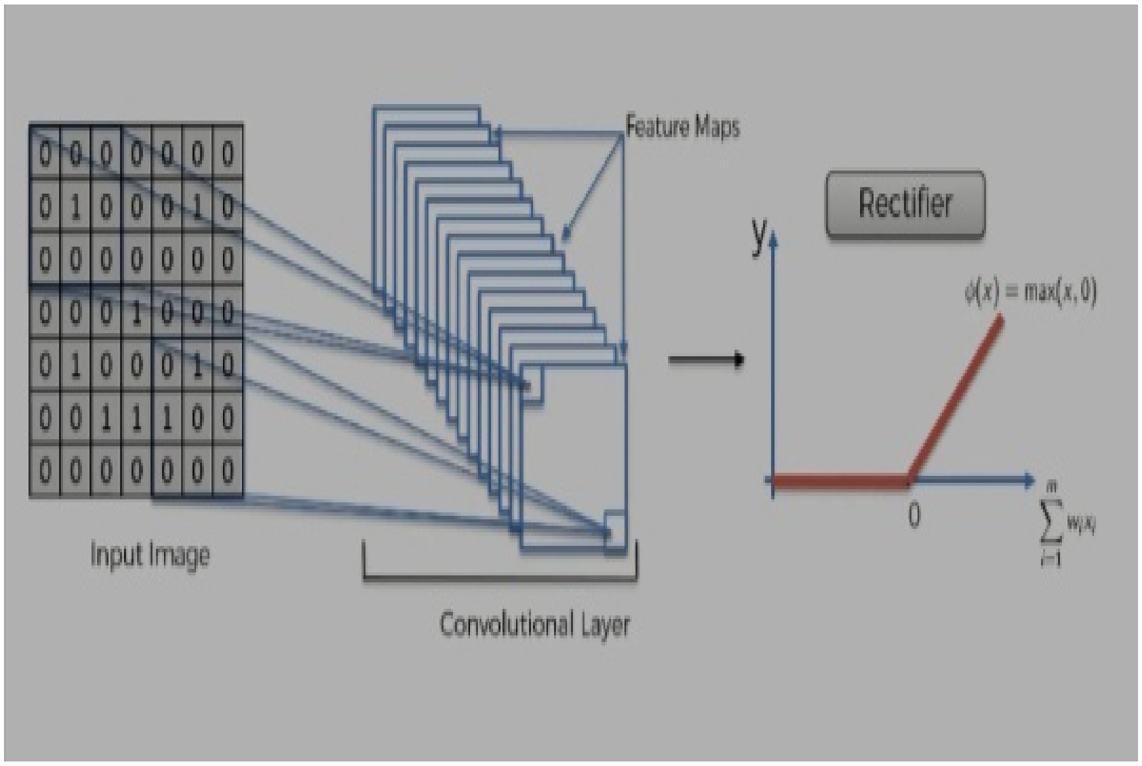


Figure 3.2.5: ReLU

4. Activation Functions

With the help of neural networks, no-linear activation functions are added between each layer. Many non-linearity functions were introduced after the first introduction of neural networks. Sigmoid functions, tanh (hyperbolic tangent) functions, and rectified Linear functions are examples of functions. The last one is the most common in modern systems and excludes sophisticated computing functions such as sigmoid and tanh functions. Neural Network Components Input Layer The input layer is first. The data will be accepted by this layer and forwarded to the remainder of the network. This layer allows feature input. It feeds the network with data from the outside world; no calculation is done here; instead, nodes simply transmit the information (features) to the hidden units.

Hidden Layer Since they are a component of the abstraction that any neural network provides, the nodes in this layer are not visible to the outside world. Any features entered through to the input layer are processed by the hidden layer in any way, with the results being sent to the output layer. The concealed layer is the name given to the second kind of layer. For a neural network, either there are one or many hidden layers. The number inside the example above is 1. In reality, hidden layers are what give neural networks their exceptional performance and intricacy. They carry out several tasks concurrently, including data transformation and automatic feature generation.

Output Layer This layer raises the knowledge that the network has acquired to the outside world. The output layer is the final kind of layer The output layer contains the answer to the issue. We receive output from the output layer after passing raw photos to the input layer.

Data science makes extensive use of the rectified unit (ReLU) functional or the category of sigmoid processes, which also includes the logistic regression model, logistic hyperbolic tangent, and arctangent function.Because they introduce non-linearities in neural networks and enable the neural networks can learn powerful operations, activation functions are helpful. A feedforward neural network might be refactored into a straightforward linear function or matrix transformation on to its input if indeed the activation functions were taken out.

By generating a weighted total and then including bias with it, the activation function determines whether a neuron should be turned on. The activation function seeks to boost a neuron's output's nonlinearity.

3.3 Technologies

1.Python

Python is currently the most widely used multi-purpose, high-level programming language.Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.Python language is being used by almost all tech-giant companies like– Google, Amazon, Facebook, Instagram, Dropbox, Uber, etc. Python Features Python's features include Easy-to-learn: Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly. Easy-to-read: Python code is more clearly defined and visible to the eyes. Easy-to-maintain: Python's source code is fairly easy-to-maintain. A broad standard library: Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh. Interactive Mode: Python has support for an interactive mode which allows interactive testing and debugging of snippets of code. Portable: Python can run on a wide variety of hardware platforms and has the same interface on all platforms. Extendable: You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient. Databases: Python provides interfaces to all major commercial databases. GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix. Scalable: Python provides a better structure and support for large programs than shell scripting. Apart from the above-mentioned features, Python has a big list of good features, few are listed below: It supports functional and structured programming methods as well as OOP. It can be used as a scripting language or can be compiled to byte-code for building large applications. It provides very high-level dynamic data types and supports dynamic type checking. IT supports automatic garbage collection. It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java. Python is an appealing scripting language for application development since it is simple to learn while being strong and diverse.

Figure 3.3.1: Python

Python's syntax and dynamic type, combined with the fact that it is interpreted, make it an ideal language for scripting and rapid application development.

Python supports a variety of programming patterns, including object-oriented, imperative, functional, and procedural programming.

Python is not meant to be used in a certain field, such as web programming. Because it can be used with online, enterprise, 3D CAD, and other applications, it is regarded as a multipurpose programming language.

Because variables are dynamically typed, we don't need to use data types to declare them, therefore we can write a=10 to assign an integer value to an integer variable.

Python speeds up development and debugging since it has a built-in debugger.

Python 2 vs. Python 3

In most programming languages, when a new version is released, it retains the functionality and syntax of the previous version, making it easier for projects to transition to the current version. However, in the case of Python, the two versions, Python 2 and Python 3, are considerably different. A list of differences between Python 2 and Python 3 are given below:
Python 2 uses print as a statement and used as print ”something”to print some string on the console. On the other hand, Python 3 uses print as a function and used as print(”something”) to print something on the console. In Python 2, the implicit string type is ASCII, whereas, in Python 3, the implicit string type is Unicode. Python 3 doesn’t contain the xrange() function of Python 2. The xrange() is the variant of range() function which returns a xrange object that works similar to Java iterator. The range() returns a list for example the function range(0,3) contains 0, 1, 2. There is also a small change made in Exception handling in Python 3. It defines a keyword as which is necessary to be used.

2. Machine Learning

As the name implies, machine learning is all about computers learning automatically without being explicitly programmed or learning without any direct human participation. This machine learning method begins with feeding them high-quality data, followed by training the machines by constructing various machine learning models utilising the data and various algorithms. The algorithms we use are determined by the type of data we have and the task we are attempting to automate. In terms of the formal definition of Machine Learning, we can state that a Machine Learning algorithm learns from experience E in relation 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, consider using a Machine Learning algorithm to play chess. Then the experience E is obtained by playing many chess games, the task T is obtained by playing chess with many players, and the performance measure P is the chance that the algorithm would win the chess game.

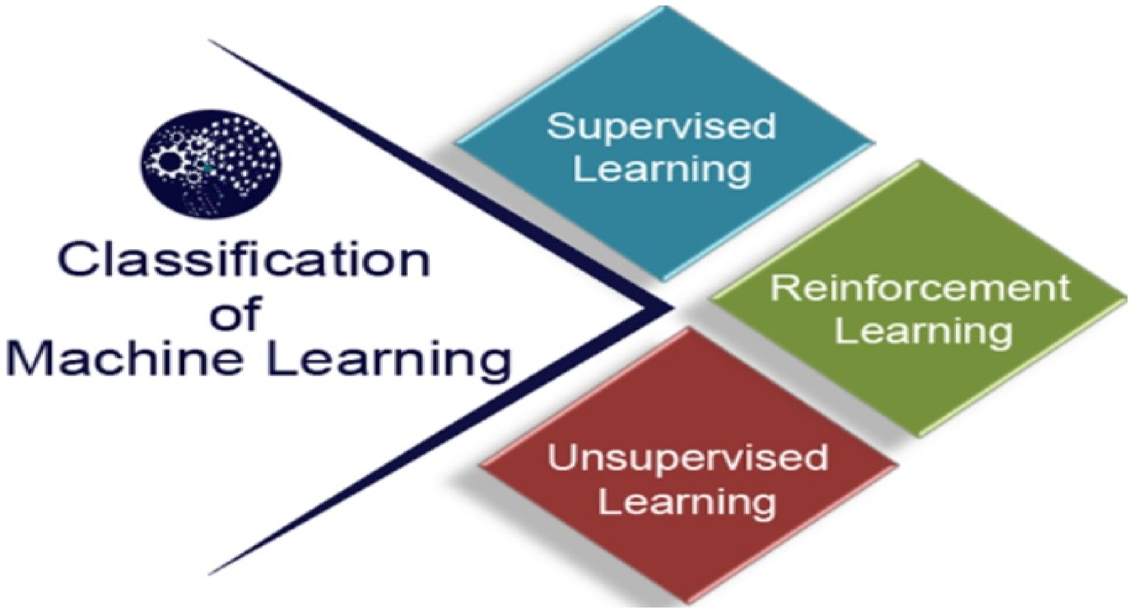


Figure 3.3.2: Classification Of Machine Learning

Different Types Of Machine Learning:

1. Supervised Machine Learning

Consider a teacher in charge of a class. The teacher already knows the proper answers, but the learning process does not end until the students do. This is what Supervised Machine Learning Algorithms are all about. The algorithm in this case learns from a training dataset and makes predictions that are then compared to the actual output values. If the forecasts are not accurate, the algorithm is tweaked until it is. This process of learning will continue until the algorithm gets the desired level of performance. Then, for any additional inputs, it may deliver the desired output values.

2. Unsupervised Machine Learning

In this situation, there is no teacher for the class, therefore the pupils must learn on their own! As a result, there is no exact answer to be learnt and no teacher for Unsupervised Machine Learning Algorithms. In this manner, the algorithm does not generate any output from the input, but instead investigates the data. The algorithm is left unsupervised in order to discover the underlying structure in the data and learn more about it.

3. Semi Supervised Machine Learning

In Semi-Supervised Machine Learning, pupils study both from their teacher and on their own. And you can tell just by looking at the name! This is a hybrid of Supervised and Unsupervised Machine Learning in which the algorithms are trained using a small amount of labelled data from Supervised Machine Learning and a larger amount of unlabeled data from Unsupervised Machine Learning. The labelled data is first used to partially train the Machine Learning Algorithm, and this partially trained model is then used to pseudo-label the remaining unlabeled data. Finally, the Machine Learning Algorithm is fully trained on labelled and pseudo-labeled data.

4. Reinforcement Machine Learning

So, here are some hypothetical pupils that, over time, learn from their own mistakes (much like in real life!). As a result, Reinforcement Machine Learning Algorithms learn the best actions through trial and error. This means that the algorithm determines the next action by learning behaviours based on its present state and maximising the reward in the future. This is accomplished by the use of reward feedback, which enables the Reinforcement Algorithm to learn which behaviours result in the greatest reward. This is referred to as a reinforcement signal.

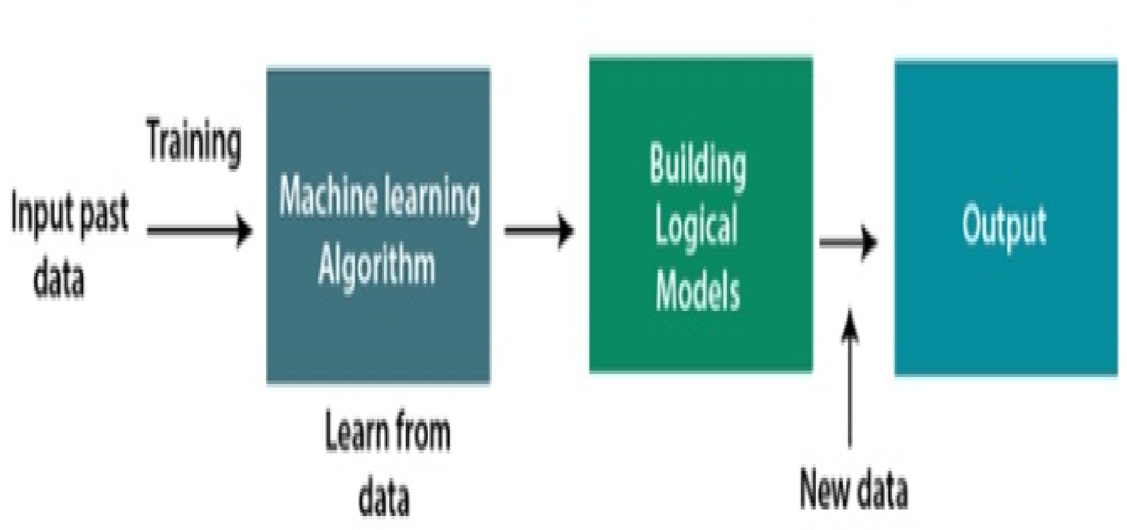


Figure 3.3.3: Machine Learning

3. Machine Learning Approaches

Supervised learning: It is a machine learning technique in which the neural network learns to generate predictions or classify data based on labelled datasets. We enter both input features as well as the target variables here. Unsupervised learning: No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself discovering hidden patterns in data or a means towards an end.

4.Deep Learning

Deep learning is a subset of machine learning that is based on the architecture of artificial neural networks. An artificial neural network, or ANN, is composed of layers of interconnected nodes known as neurons that collaborate to process and learn from input data.

An input layer and one or more hidden layers are connected one after the other in a fully connected Deep neural network. Each neuron receives information from neurons in the previous layer or the input layer. The output of one neuron becomes the input to other neurons in the network's next layer, and so on until the last layer generates the network's output. The neural network's layers change the incoming data through a series of nonlinear transformations, allowing the network to learn complex representations of the input data.Today Deep learning has become one of the most popular and visible areas of machine learning, due to its success in a variety of applications, such as computer vision, natural language processing, and Reinforcement learning.

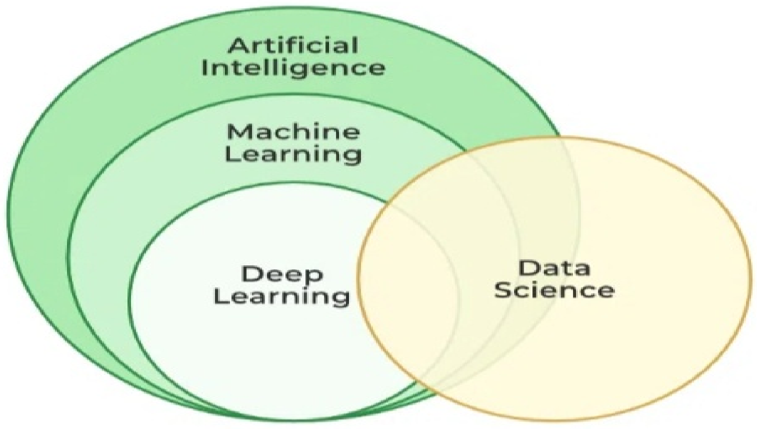


Figure 3.3.4: Representation Of Deep Learning

Deep learning can be used for both supervised and unsupervised machine learning, as well as reinforcement learning. It processes these in a variety of ways.
Supervised Machine Learning: It is a machine learning technique in which the neural network learns to generate predictions or classify data based on labelled datasets. We enter both input features as well

as the target variables here. Backpropagation is the method through which a neural network learns to generate predictions based on the cost or mistake that results from the difference between the projected and real goal. Deep learning methods such as Convolutional neural networks and Recurrent neural networks are used for a variety of supervised tasks such as picture classification and recognition, sentiment analysis, language translations, and so on.

Unsupervised Machine Learning:

Unsupervised machine learning is a type of machine learning technique in which a neural network learns to detect patterns or cluster datasets using unlabeled datasets. There are no target variables in this case. while the machine must discover hidden patterns or relationships within the datasets. Unsupervised tasks like as clustering, dimensionality reduction, and anomaly detection are handled by deep learning techniques such as autoencoders and generative models.

Reinforcement Machine Learning:

Reinforcement The machine learning process in which an agent learns to make decisions in an environment to maximise a reward signal is known as machine learning. The agent interacts with the environment by acting and monitoring the results. Deep learning can be used to learn policies, or a collection of actions, that maximise the total reward over time. Deep reinforcement learning techniques, such as Deep Q networks and Deep Deterministic Policy Gradient (DDPG), are used to reinforce tasks like as robotics and gaming.

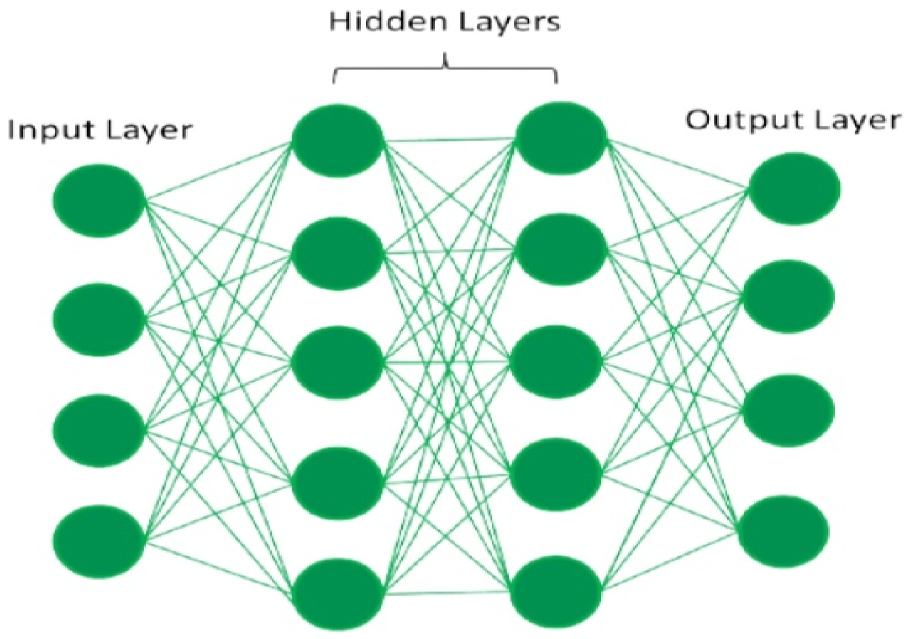


Figure 3.3.5: Fully Connected Neural Network

In artificial neural networks, artificial neurons, also known as units, can be found. These artificial neurons, which are stacked in a succession of layers, make up the entire Artificial Neural Network. Whether a layer includes a dozen units or millions of units, the complexity of neural networks are determined by the complexities of the underlying patterns in the dataset. An Artificial Neural Network typically contains an input layer, an output layer, and hidden layers. The input layer gets information from the outside world that the neural network must analyse or learn. An input layer and one or more hidden layers are connected one after the other in a fully connected artificial neural network. Each neuron receives information from neurons in the previous layer or the input layer. The output of one neuron becomes the input to other neurons in the network's next layer, and so on until the last layer generates the network's output. The data is then turned into valuable data for the output layer after travelling through one or more hidden layers. Finally, the output layer generates output in the form of an artificial neural network's reaction to the input data.

3.4 UML Diagrams

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML. The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.Complex applications need collaboration and planning from multiple teams and hence require a clear and concise way to communicate amongst them. Businessmen do not understand code. So UML becomes essential to communicate with non programmers essential requirements, functionalities and processes of the system. A lot of time is saved down the line when teams are able to visualize processes, user interactions and static structure of the system. Since it is a general-purpose modeling language, it can be utilized by all the modelers. UML came into existence after the introduction of object-oriented concepts to systemize and consolidate the object-oriented development, due to the absence of standard methods at that time. The UML diagrams are made for business users, developers, ordinary people, or anyone who is looking forward to understand the system, such that the system can be software or non-software. Thus it can be concluded that the UML is a simple modeling approach that is used to model all the practical systems. The UML has the following features:

It is a generalized modeling language. It is distinct from other programming languages like C++, Python, etc. It is interrelated to object-oriented analysis and design. It is used to visualize the workflow of the system. It is a pictorial language, used to generate powerful modeling artifacts. UML is linked with object oriented design and analysis. UML makes the use of elements and forms associations between them to form diagrams. Diagrams in UML can be broadly classified as: The Object Management Group (OMG) is an association of several companies that controls the open standard UML. The OMG was established to build an open standard that mainly supports the interoperability of object-oriented systems. It is not restricted within the boundaries, but it can also be utilized for modeling the non-software systems. The OMG is best recognized for the Common Object Request Broker Architecture (CORBA) standards.The goal of UML is to provide a standard notation that can be used by all object-oriented methods and to select and integrate the best elements of precursor notations. UML has been designed for a broad range of applications. Hence, it provides constructs for a broad range of systems and activities (e.g., distributed systems, analysis, system design and deployment).

3.4.1 Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.In Unified Modeling Language (UML), systems are presented at different levels of detail to show a specific perspective in the system's design. Use case diagrams are considered UML diagrams.A use case diagram is a visual representation of the different ways and possible scenarios of using a system. It illustrates how a user will perform actions and interact with a particular system, such as a website or an app.Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The main purpose of a use case diagram is to portray the dynamic aspect of a system. It accumulates the system's requirement, which includes both internal as well as external influences. It invokes persons, use cases, and several things that invoke the actors and elements accountable for the implementation of use case diagrams. It represents how an entity from the external environment can interact with a part of the system.

Following are the purposes of a use case diagram given below:

It gathers the system's needs. It depicts the external view of the system. It recognizes the internal as well as external factors that influence the system. It represents the interaction between the actors. It is essential to analyze the whole system before starting with drawing a use case diagram, and then the system's functionalities are found. And once every single functionality is identified, they are then transformed into the use cases to be used in the use case diagram.After that, we will enlist the actors that will interact with the system. The actors are the person or a thing that invokes the functionality of a system. It may be a system or a private entity, such that it requires an entity to be pertinent to the functionalities of the system to which it is going to interact.

Once both the actors and use cases are enlisted, the relation between the actor and use case/ system is inspected. It identifies the no of times an actor communicates with the system. Basically, an actor can interact multiple times with a use case or system at a particular instance of time.

Following are some rules that must be followed while drawing a use case diagram:
A pertinent and meaningful name should be assigned to the actor or a use case of a system. The communication of an actor with a use case must be defined in an understandable way. Specified notations to be used as and when required. The most significant interactions should be represented among the multiple no of interactions between the use case and actors.

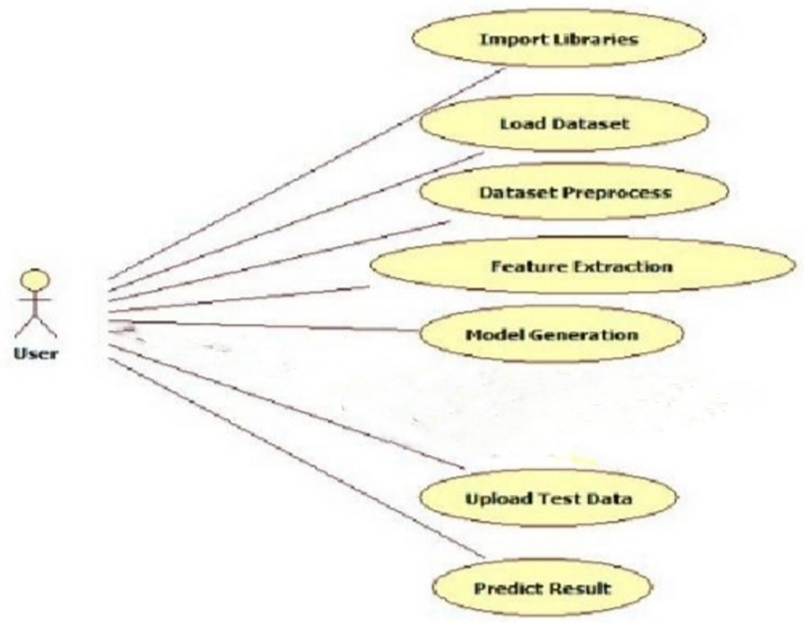


Figure 3.4.1: Use Case Diagram Of Video Streaming Caption Generator

3.5 Software and Hardware Requirements

Hardware

i) Operating System: Windows 11

Windows 11 is a graphical operating system developed by Microsoft. It allows users to view and store files, run the software, play games, watch videos, and provides a way to connect to the internet.

ii) Processor: Intel I5

Intel I5 Core i5 Developed and manufactured by Intel, the Core i5 is a computer processor, available as dual-core or quad-core. It can be used in both desktop and laptop computers.

iii) Hard disk: 1TB

A hard disk drive (HDD), hard disk, hard drive, or fixed disk [b] is an electro-mechanical data storage device that stores and retrieves digital data using magnetic storage.

Software

1) python

Python is a high-level, general-purpose and a very popular programming language. Python programming language (latest Python 3) is being used in web development, Machine Learning applications It provides very high-level dynamic data types and supports dynamic type checking. IT supports automatic garbage collection. It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java. Python is an appealing scripting language for application development since it is simple to learn while being strong and diverse.

2) Google Collab

olaboratory, or Colab for short, is a Google Research product, which allows developers to write and execute Python code through their browser. Google Colab is an excellent tool for deep learning tasks. It is a hosted Jupyter notebook that requires no setup and has an excellent free version, which gives free access to Google computing resources such as GPUs and TPUs..

3) Kaggle Notebook

Run Data Science Machine Learning Code Online — Kaggle Kaggle Notebooks are a computational environment that enables reproducible and collaborative analysis.

4) Libraries Used:

i)numpy.

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python. In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.

ii).pandas.

Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance productivity for users.

iii).OpenCV.

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as Numpy which is a highly optimized library for numerical operations, then the number of weapons increases in your Arsenal i.e whatever operations one can do in Numpy can be combined with OpenCV.

NumPy is a Python library used for working with arrays.It also has functions for working in domain of linear algebra, fourier transform, and matrices.Pandas is a Python library used for working with data sets.It has functions for analyzing, cleaning, exploring, and manipulating data. OpenCV is created to implement various operations including recognising and detecting faces, analysing tasks in videos, identifying objects, recording camera movements, tracking moving objects.

4. Methodology

4.1 Architecture

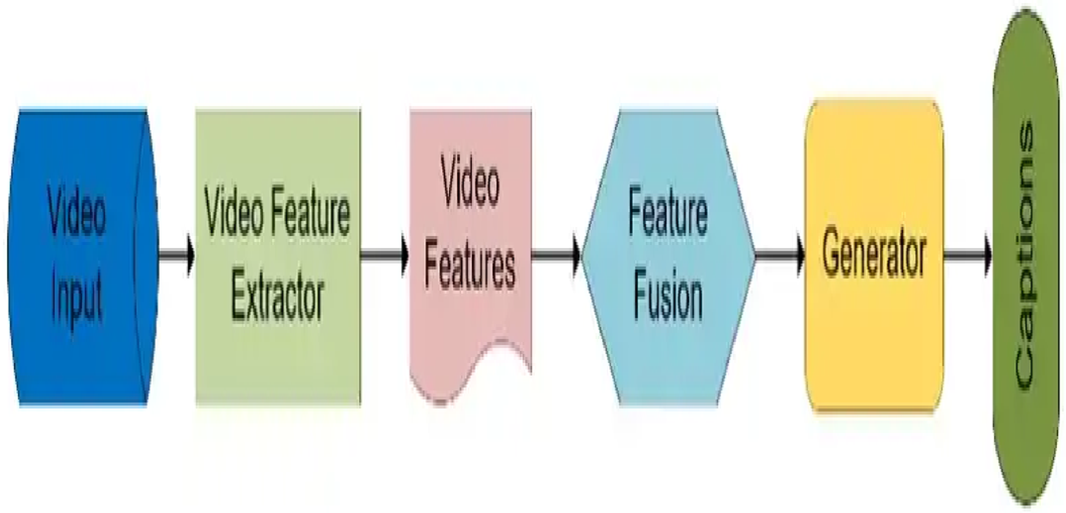


Figure 4.1.1: Architecture

In the Figure 4.1 Initially the video is given as input to the model then video features are extracted through the CNN Algorithm the CNN algorithm is used for image preprocessing where the unwanted data will be removed to get output more accurately and by using the generator the captions will be generated for the given input video.

4.2 Modules

Modules are important to have a precise overview on the development of the project process so that while execution clarity of the next step is maintained.Video Streaming Caption Generator has following modules

i) DataSet Collection

ii) DataSet Preprocessing

iii) Feature Extraction

iV) Model Generation

i) DataSet Collection : We are using UCF101 dataset, that contains videos and videos related captions and it consists of 13,320 video clips, which are classified into 101 categories. These 101 categories can be classified into 5 types (Body motion, Human-human interactions, Human-object interactions, Playing musical instruments and Sports).

ii) DataSet Preprocessing : Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.Data preprocessing is a Data Mining method that entails converting raw data into a format that can be understood. Real-world data is frequently inadequate, inconsistent, and/or lacking in specific activities or trends, as well as including numerous inaccuracies. This might result in low-quality data collection and, as a result, low-quality models based on that data. Preprocessing data is a method of resolving such problems. Machines do not comprehend free text, image, or video data; instead, they comprehend 1s and 0s.So putting on a slideshow of all our photographs and expecting our machine learning model to learn from it is probably not going to be adequate. Data Preprocessing is the step in any Machine Learning process in which the data is changed, or encoded, to make it easier for the machine to parse it.Provides several common utility functions and transformer classes to change raw feature vectors into a representation that is more suitable for the downstream estimators.n practice we often ignore the shape of the distribution and just transform the data to center it by removing the mean value of each feature, then scale it by dividing non-constant features by their standard deviation. A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

iii) Feature Extraction : By using the CNN algorithm the feature extraction will be done in images which are present in the video are captured to identify the important features. Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. A characteristic of these large data sets is a large number of variables that require a lot of computing resources to process. Feature extraction is the name for methods that select and /or combine variables into features, effectively reducing the amount of data that must be processed, while still accurately and completely describing the original data set.The process of feature extraction is useful when you need to reduce the number of resources needed for processing without losing important or relevant information. Feature extraction can also reduce the amount of redundant data for a given analysis. Also, the reduction of the data and the machine's efforts in building variable combinations (features) facilitate the speed of learning and generalization steps in the machine learning process.A Convolutional Neural Network (CNN) is a type of deep learning algorithm that is particularly well-suited for image recognition and processing tasks. It is made up of numerous layers, including convolutional layers, pooling layers, and fully linked layers. The fundamental component of a CNN is the convolutional layers, which apply filters to the input image to extract characteristics like as edges, textures, and forms. The convolutional layers' output is then passed through pooling layers, which are used to down-sample the feature maps, reducing the spatial dimensions while retaining the most important information. The pooling layers' output is then passed through one or more fully connected layers, which are used to predict or classify the image. CNNs are trained on a huge dataset of labelled images, with the network learning to recognize patterns and attributes associated with specific objects or classes. A CNN that has been trained can be used to categorize new images or extract features for use in other applications such as object detection or image segmentation

iv) Model Generation :

The model is created after it has been trained using the CNN algorithm. The dataset will be divided into two sections for model generation: one for training and one for testing. 70 percent of the data is used for training, 30 percent for testing, and the suitable top 5 captions will be made for the provided input video. The caption with the highest accuracy will be generated first, and the captions will be generated based on the accuracy.

5. Implementation

Below is the important part of python coding which properly arranges the taken data set to train into train data and test data so that it can be given as input to the deep learning methodologies.

5.1 Code

```
!python --version


!pip install -q imageio

!pip install -q opencv-python

##!pip install -q git+https://github.com/tensorflow/docs


!pip install -q git+https://github.com/MJAHMADEE/docs


#@title Import the necessary modules

# TensorFlow and TF-Hub modules.

from absl import logging


import tensorflow as tf

import tensorflow_hub as hub

from tensorflow_docs.vis import embed

logging.set_verbosity(logging.ERROR)


# Some modules to help with reading the UCF101 dataset.

import random

import re

import os

import tempfile
import ssl
import cv2
import numpy as np

# Some modules to display an animation using imageio.
import imageio
from IPython import display
from urllib import request  # requires python3
UCF_ROOT = "https://www.crcv.ucf.edu/THUMOS14/UCF101/UCF101/"
_VIDEO_LIST = None
_CACHE_DIR = tempfile.mkdtemp()
unverified_context = ssl._create_unverified_context()

def list_ucf_videos():
  """Lists videos available in UCF101 dataset."""
  global _VIDEO_LIST
  if not _VIDEO_LIST:
    index = request.urlopen(UCF_ROOT,
    context=unverified_context).read().decode("utf-8")
    videos = re.findall("(v_[\w_]+\.avi)", index)
    _VIDEO_LIST = sorted(set(videos))
  return list(_VIDEO_LIST)

def fetch_ucf_video(video):
  """Fetchs a video and cache into local filesystem."""
  cache_path = os.path.join(_CACHE_DIR, video)
  if not os.path.exists(cache_path):
    urlpath = request.urljoin(UCF_ROOT, video)
    print("Fetching %s => %s" % (urlpath, cache_path))
    data = request.urlopen(urlpath, context=unverified_context).read()
    open(cache_path, "wb").write(data)
  return cache_path
# Utilities to open video files using CV2
def crop_center_square(frame):
  y, x = frame.shape[0:2]
  min_dim = min(y, x)
  start_x = (x // 2) - (min_dim // 2)
  start_y = (y // 2) - (min_dim // 2)
  return frame[start_y:start_y+min_dim,start_x:start_x+min_dim]

def load_video(path, max_frames=0, resize=(224, 224)):
  cap = cv2.VideoCapture(path)
  frames = []
  try:
    while True:
      ret, frame = cap.read()
      if not ret:
        break
      frame = crop_center_square(frame)
      frame = cv2.resize(frame, resize)
      frame = frame[:, :, [2, 1, 0]]
      frames.append(frame)

      if len(frames) == max_frames:
        break
  finally:
    cap.release()
  return np.array(frames) / 255.0
def to_gif(images):
  converted_images = np.clip(images * 255, 0, 255).astype(np.uint8)
  imageio.mimsave('./animation.gif', converted_images, fps=25)
  return embed.embed_file('./animation.gif')

KINETICS_URL = "https://raw.githubusercontent.com/deepmind/kinetics-
i3d/master/data/label_map.txt"
with request.urlopen(KINETICS_URL) as obj:
  labels = [line.decode("utf-8").strip() for line in obj.readlines()]
print("Found %d labels." % len(labels))

# Get the list of videos in the dataset.
ucf_videos = list_ucf_videos()

categories = {}
for video in ucf_videos:
  category = video[2:-12]
  if category not in categories:
    categories[category] = []
  categories[category].append(video)
print("Found %d videos in %d categories." % (len(ucf_videos), len(categories)))

for category, sequences in categories.items():
  summary = ", ".join(sequences[:2])
  #print("%-20s %4d videos (%s, ...)" % (category, len(sequences), summary))


i3d = hub.load("/kaggle/input/i3d
kinetics/tensorflow1/400/1").signatures['default']

def predict(sample_video):
  # Add a batch axis to the sample video.
  model_input = tf.constant(sample_video, dtype=tf.float32)[tf.newaxis, ...]

  logits = i3d(model_input)['default'][0]
  probabilities = tf.nn.softmax(logits)

  print("Top 5 actions:")
  for i in np.argsort(probabilities)[::-1][:5]:
    print(f"  {labels[i]:22}: {probabilities[i] * 100:5.2f}


video_path = "/kaggle/input/actionrecogsamplevideos/v_PlayingFlute_g01_c01.avi"
sample_video = load_video(video_path)[:100]
sample_video.shape
predict(sample_video)
to_gif(sample_video)
```

5.2 Testing The Model

The first step is to upload the video from the dataset . Following that, it will enter several modules and output the related description for a user-provided input, with the captions with the highest accuracy appearing first. For the provided input video, five outputs will be generated, and the caption with the highest accuracy is regarded the best caption.

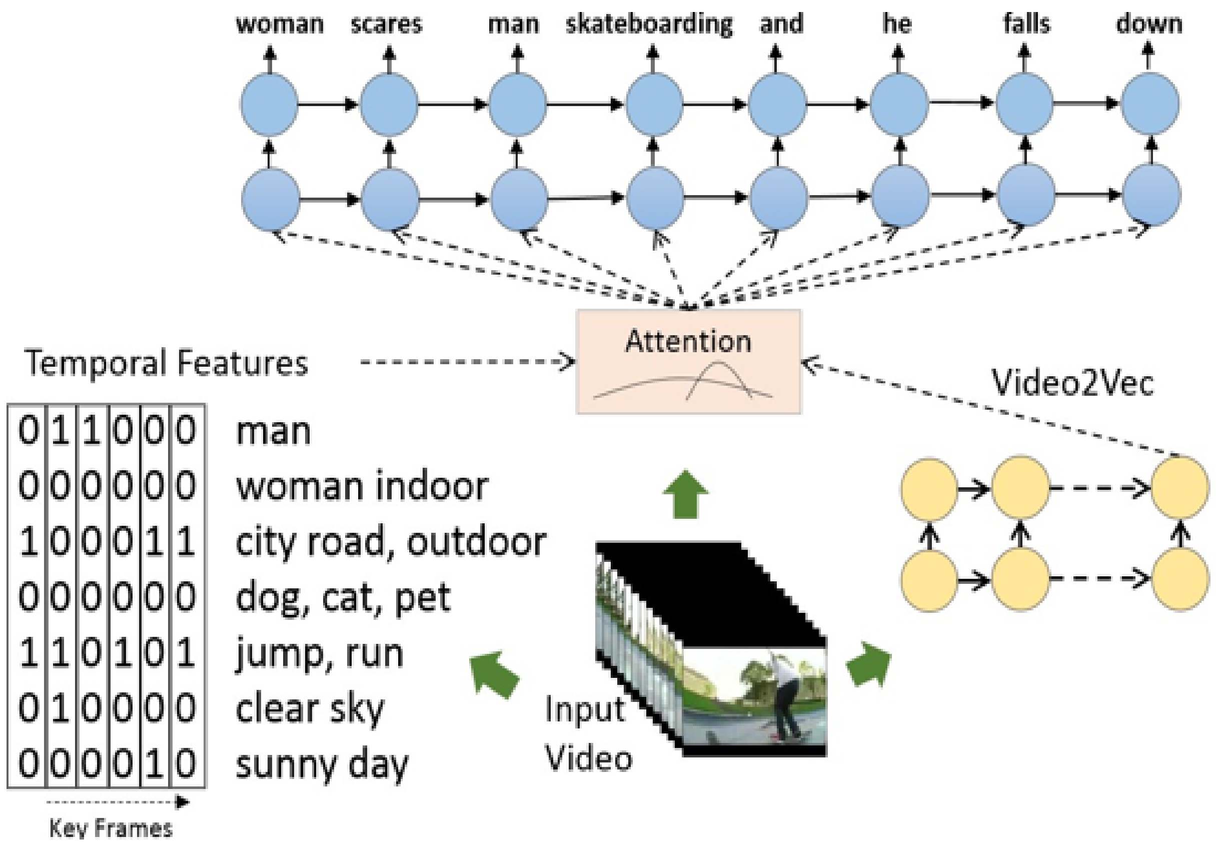


Figure 5.2.1: Overview Of Model For Video Captioning

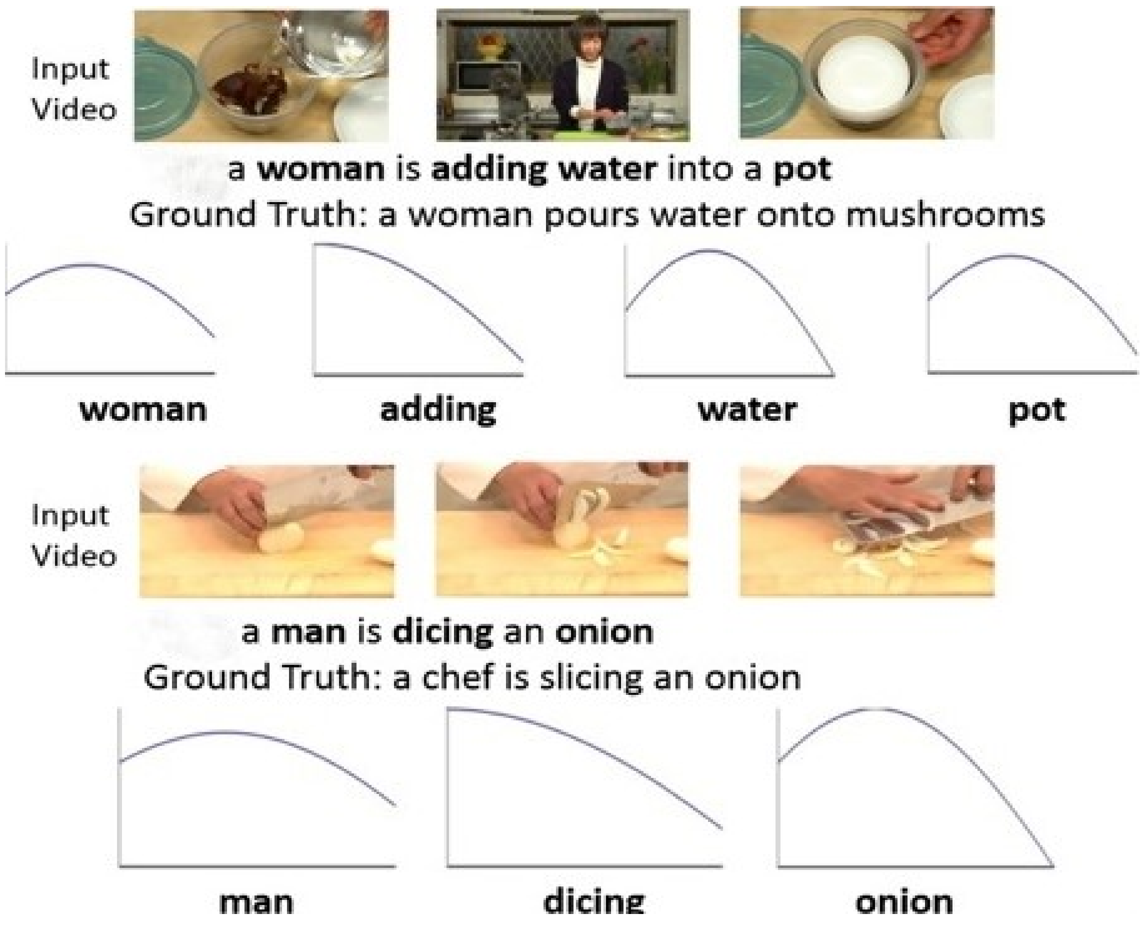
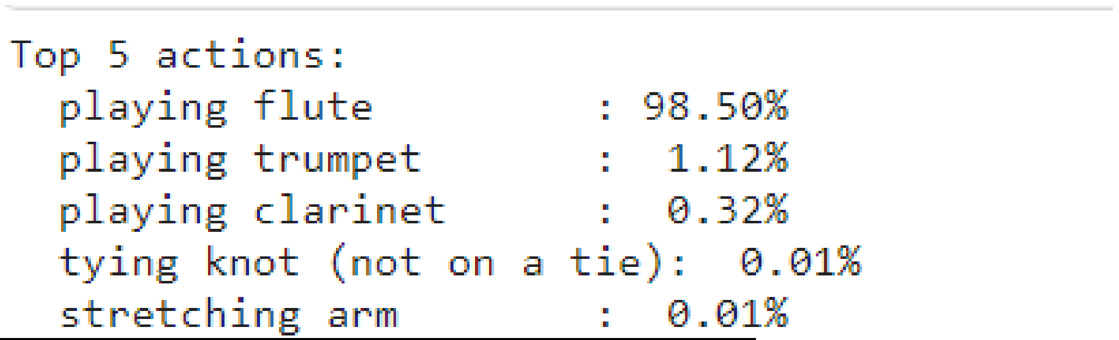

Input Video
a woman is adding water into a pot
Ground Truth: a woman pours water onto mushrooms
woman
adding
water
pot
Input Video
a man is dicing an onion
Ground Truth: a chef is slicing an onion
man
dicing
onion

Figure 5.2.2: Testing the Model

5.3 Results

Captions for the provided input video will be created. For each video, several captions will be present in the dataset and the top five actions will be displayed, with the caption with the highest accuracy appearing first.



```
Top 5 actions:
  playing flute          : 98.50%
  playing trumpet        :  1.12%
  playing clarinet       :  0.32%
  tying knot (not on a tie):  0.01%
  stretching arm         :  0.01%
```

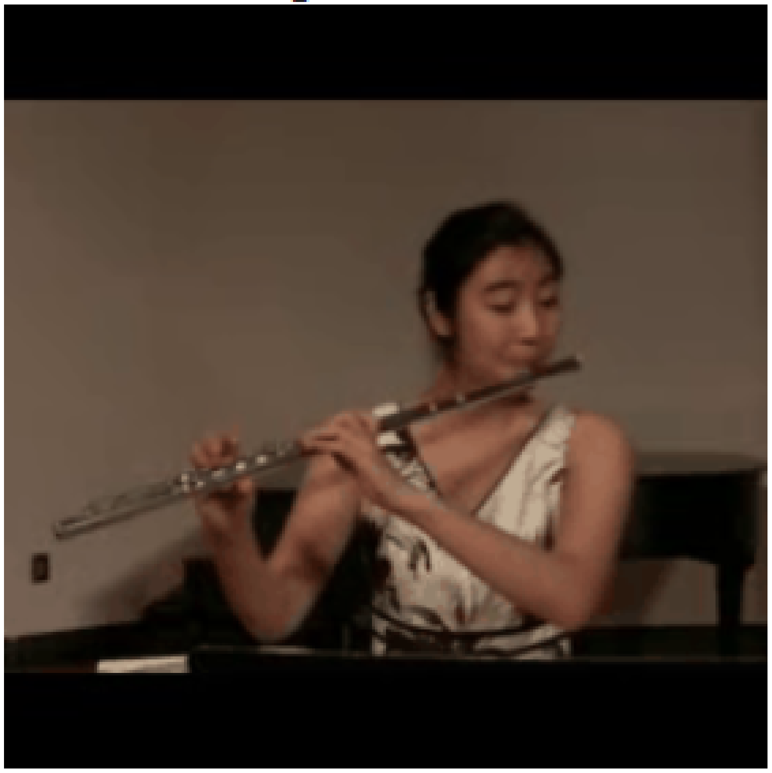


Figure 5.3.1: Playing Flute

```
Top 5 actions:
  playing cricket         : 99.64%
  golf putting            :  0.15%
  robot dancing           :  0.09%
  shooting goal (soccer):  0.05%
  marching                :  0.01%
```

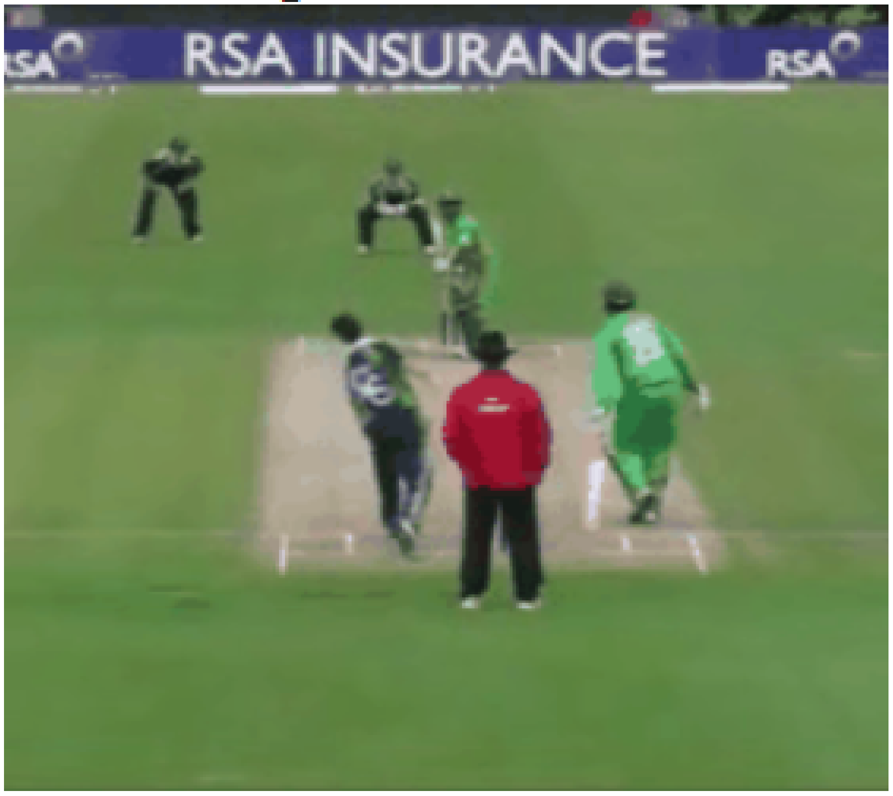


Figure 5.3.2: Playing Cricket

```
Top 5 actions:
  crawling baby          : 99.99%
  laughing               :  0.01%
  shaking head           :  0.00%
  pushing car            :  0.00%
  baby waking up         :  0.00%
```

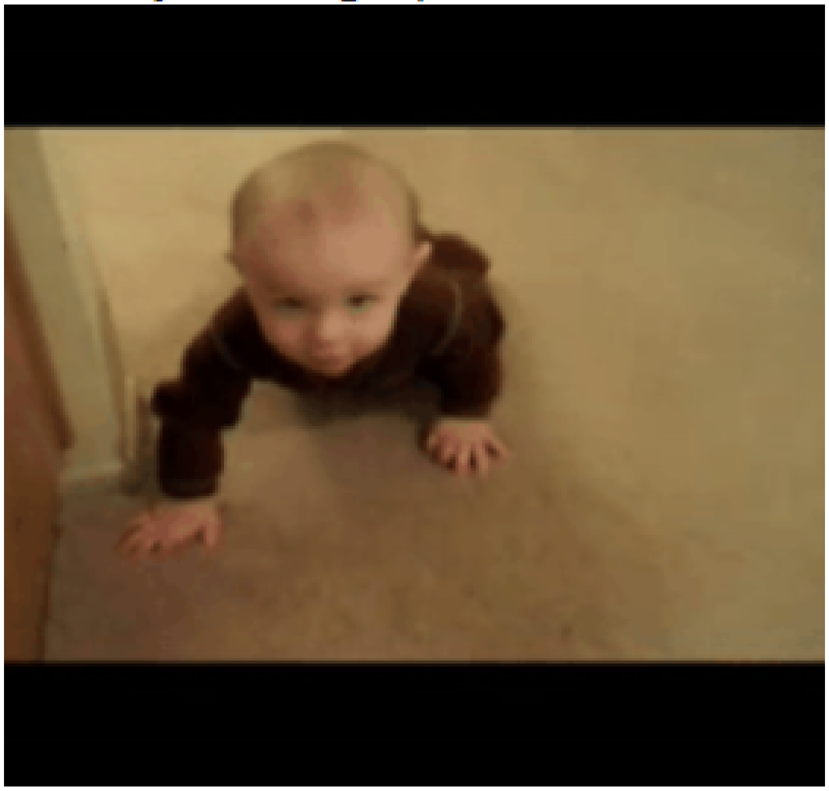
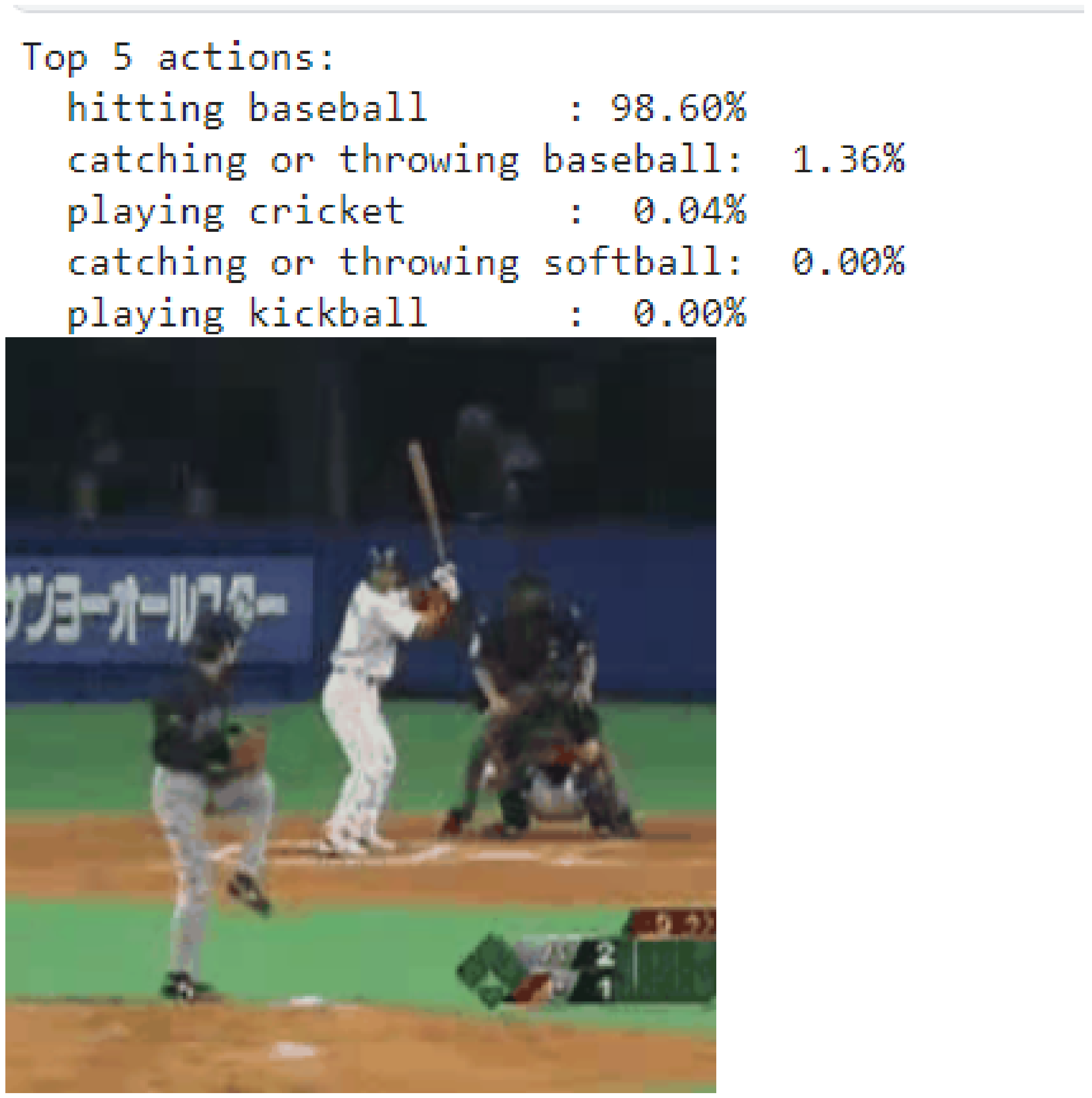
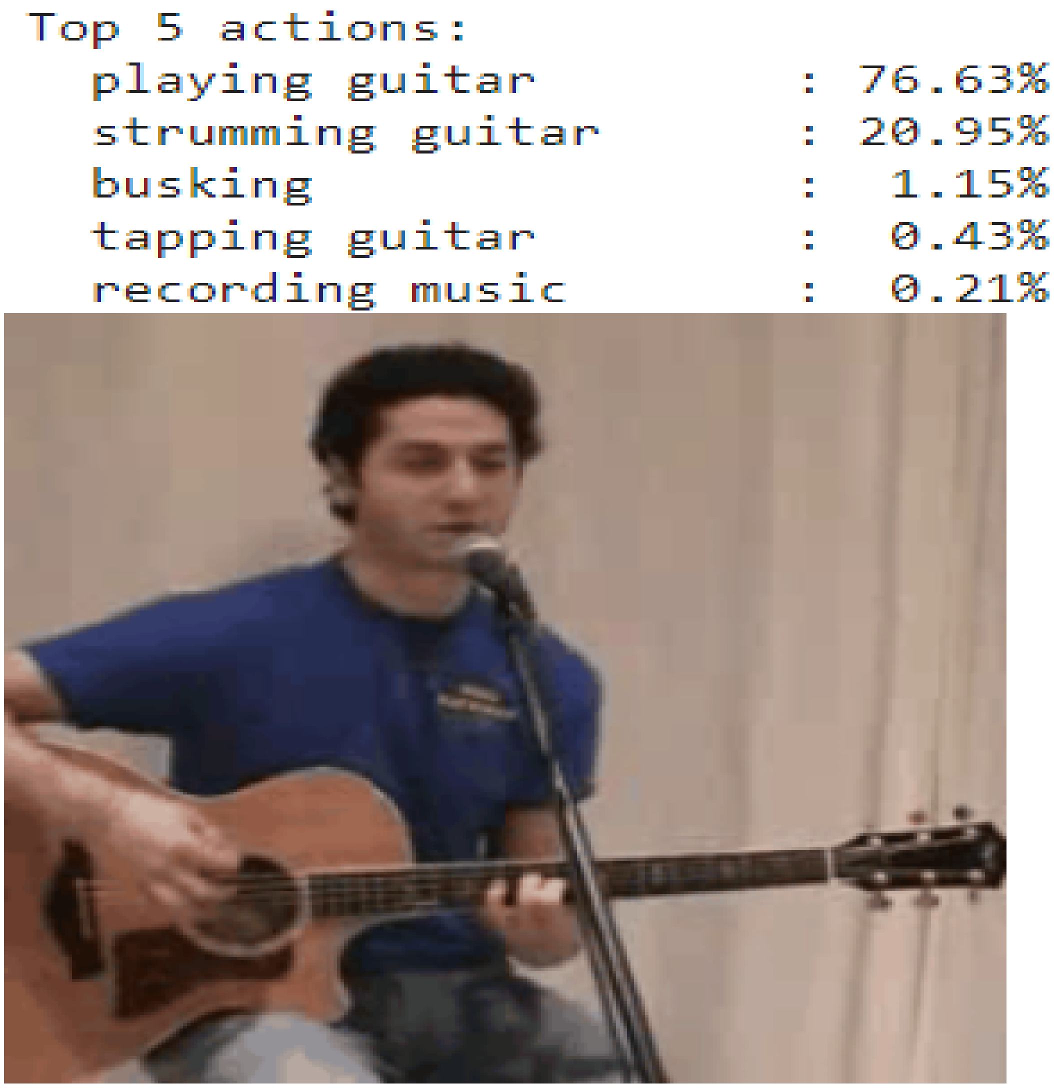


Figure 5.3.3: Baby Crawling



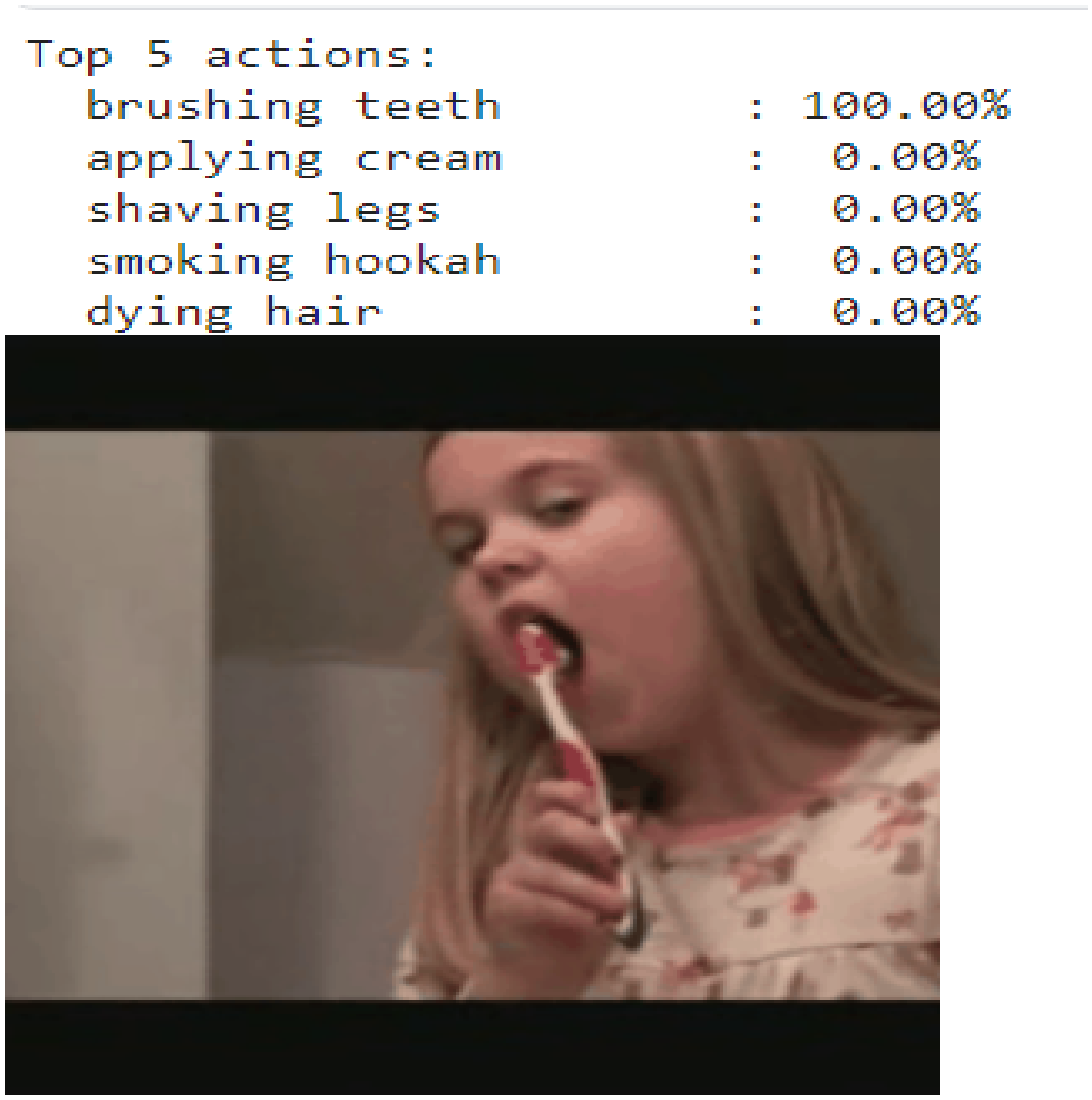
```
Top 5 actions:
  hitting baseball        : 98.60%
  catching or throwing baseball:  1.36%
  playing cricket         :  0.04%
  catching or throwing softball:  0.00%
  playing kickball        :  0.00%
```

Figure 5.3.4: Hitting Baseball



```
Top 5 actions:
  playing guitar            : 76.63%
  strumming guitar          : 20.95%
  busking                   :  1.15%
  tapping guitar            :  0.43%
  recording music           :  0.21%
```

Figure 5.3.5: Playing Guitar



```
Top 5 actions:
  brushing teeth          : 100.00%
  applying cream          :   0.00%
  shaving legs            :   0.00%
  smoking hookah          :   0.00%
  dying hair              :   0.00%
```

Figure 5.3.6: Brushing Teeth

6. Conclusion and Future Scope

A reliable, accurate, and real-time video captioning method can be used in many applications. Video captioning systems can be used as an important part of Assistive Technologies that would help people with hearing impairments and video Captions can be used as part of recommendation systems in many applications.

Instead of using the features provided, extract more features using models such as I3D that are specifically built for movies. Adding a user interface to make things more appealing and distributing it to a platform. To train for longer videos, embedding layers and attention blocks are being added.

REFERENCES

[1] J. Zhang and Y. Peng, .Object-aware aggregation with bidirectional temporal graph for video captioning", Proc. IEEE Conf. Comput. Vis. Pattern Recognit., pp. 8327-8336, Jun. 2019.

[2] Li L, Gong B (2019 Jan 7) End-to-end video captioning with multitasking reinforcement learning. In: 2019 IEEE winter conference on applications of computer vision (WACV) (pp. 339-348). IEEE

[3] J. Mun, L. Yang, Z. Ren, N. Xu and B. Han, "Streamlined dense video captioning", Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 6588-6597, Jun. 2019.

[4] S.H. Abdulhussain, A. Rahman Ramli, B.M. Mahmmod, M. Iqbal Saripan, S.A.R. Al-Haddad, T. Baker, et al., .A fast feature extraction algorithm for image and video processing", International Joint Conference on Neural Networks (IJCNN), pp. 1-8, 2019.

[5] Himanshu Sharma, Manmohan Agrahari, Sujeet Kumar Singh, Mohd Firoj and Ravi Kumar Mishra, Ïmage Captioning: A Comprehensive Survey", 2020.

[6] Jin T, Li Y, Zhang Z (2019) Recurrent convolutional video captioning with global and local attention. Neurocomputing 370:118–127

[7] N. Aafaq, A. Mian, W. Liu, S. Z. Gilani and M. Shah, "Video description: A survey of methods datasets and evaluation metrics", ACM Comput. Surv., vol. 52, no. 6, pp. 1-37, Jan. 2020.