KLE Society's
KLE Technological University, Hubballi.

A Capstone Project Report

on

Image Based Auto Captioning by Face Recognition and Object Detection

submitted in partial fulfillment of the requirement for the degree of

Bachelor of Engineering

in

Computer Science and Engineering

Submitted by

Sougat Paul	01FE17BCS208
Sumegh Madawale	01FE15BCS209
Usha Kamble	01FE18BCS242
Rohit Kulkarni	01FE18BCS177

Under the guidance of
Ms. Preeti T and Mr. Mallikarjun Akki

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Hubballi – 580 031

2021 -22

KLE Society's
KLE Technological University, Hubballi.

2021 - 2022

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that Minor Project titled Image Based Auto Captioning by Face Recognition and Object Detection is a bonafied work carried out by the student team comprising of Sougat Paul 01FE17BCS208, Sumegh Madawale 01FE15BCS209, Usha Kamble 01FE17BCS242, Rohit Kulkarni 01FE18BCS177 for partial fulfillment of completion of eighth semester B.E. in Computer Science and Engineering during the academic year 2021-22.

Guide(s)

Ms. Preeti T
Mr. Mallikarjun Akki

Head, SoCSE

Dr. Meena S. M

Viva -Voce:

Name of the Examiners	Signature with date
1.	
2.	

ABSTRACT

We have focused in on significant learning ways of managing face recognizable proof and affirmation and thing area and affirmation. This investigation has essentially based on setting up the cerebrum associations or various models with a sufficient number of proportions of data so it achieves positive results. Starting with the essentials of mind network where a neuron, the humblest unit in significant learning field, is portrayed and sorted out, I have raised the investigation to the place where I could see the substance of an individual or an article in an image. First the cerebrum networks have been introduced in this assessment and a short time later planning of the mind networks has been achieved with both the CPU and the GPU. In an estimation returned to grid kind of expansion, the usage of different GPUs has been made close by CUDA part and cuBLAS library. Usage of face affirmation has been completed using the pre-arranged model Facenet and Deep Convolutional Neural Networks. Directly following separating cerebrum associations and its facial affirmation application, various systems of significant learning have been empowered. I have used the library OpenCV close by significant learning ways of managing do confront affirmation, Image selection and YOLO Object Detection and Recognition with the tensor stream and keras environment maintained by boa constrictor for python.

Picture subtitling is to naturally portray a picture with a sentence, which is a subject interfacing PC vision and normal language handling. Research on picture inscribing has incredible effect on assist outwardly weakened individuals with understanding their environmental elements, and it has possible advantages for the sentence-level photograph association. Early work commonly handled this undertaking by recovery strategies or layout techniques. Current strategies were principally founded on a blend of Convolutional Neural Networks (CNN) what's more, Recurrent Neural Networks (RNN). Nonetheless, creating precise and clear subtitles stays a difficult assignment. Exact subtitles allude to sentences steady with the visual substance, and engaging subtitles allude to those with different portrayals as opposed to plain normal sentences. By and large, the vision model is expected to encode the setting extensively and the language model is expected to communicate the visual portrayal into a clear sentence reliably. Also, the preparation methodology additionally influences the presentation.

Acknowledgement

We would like to thank our faculty and management for their professional guidance towards the completion of the project work. We take this opportunity to thank Dr. Ashok Shettar, Vice-Chancellor, Dr. N.H Ayachit, Registrar, and Dr. P.G Tewari, Dean Academics, KLE Technological University, Hubballi, for their vision and support.

We also take this opportunity to thank Dr. Meena S. M, Professor and Head, SoCSE for having provided us direction and facilitated for enhancement of skills and academic growth.

We thank our guide Ms. Preeti T, Assistant Professor, SoCSE for the constant guidance during interaction and reviews.

We extend our acknowledgement to the reviewers for critical suggestions and inputs. We also thank Project Co-ordinator Dr. Sujatha C. and faculty in-charges for their support during the course of completion.

We express gratitude to our beloved parents for constant encouragement and support.

Sougat Paul - 01FE17BCS208
Sumegh Madawale - 01FE15BCS209
Usha Kamble - 01FE18BCS242
Rohit Kulkarni - 01FE18BCS177

CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

The advancement of the product framework or the PC framework that would lead assorted errands with the human knowledge is fundamentally known as Artificial Intelligence. At the end of the day, Artificial Intelligence is the method involved with mirroring smart way of behaving in machines. Different instances of these ways of behaving incorporate visual understanding, discourse distinguishing proof or assignments including decision making and so on The most difficult aspect of counterfeit savvy is to address the pace of precision and proficiency of human cerebrum in its occupation. People are creating unfathomable measure of information consistently and are equipped for getting every part of it, that the development of man-made reasoning is sprouting with it. Like human gain from encounters and slip-ups, similarly a machine with powerful man-made reasoning will gain as a matter of fact and preparing yet not tantamount to human cerebrum. [1]

AI is a type of artificial intelligence in which machines train themselves by learning and developing without the need for human intervention or fresh programming. Profound Learning is also a sort of AI that prepares the machine to learn, train, comprehend, and face this current reality in terms of concept order, allowing the machine or PC to absorb perplexing ideas by creating them out of simpler ones. On the grounds that the actual PC merge the authority for a fact, there is no requirement for human assistance to physically operate the PC to particularize all of the information requested by the PC.Profound Learning is a self-administered, self-teaching framework that uses the data that has already been stored in it to perform more calculations in order to find alternative examples. These examples are used to develop future predictions regarding new information that the PC has never seen before. For example, when the organization was taken care of with a large number of such photos having a comparable object or creature, one could build a profound learning calculation to distinguish an item, like as a table or a glass, or a creature, such as an elephant, from a picture.The organization will then form different examples by arranging and bunching the picture information which further illuminate an anticipated model that is equipped for foreseeing the exact outcomes subsequent to taking a gander at the new arrangement of pictures once more. Profound learning calculations play out this undertaking of foreseeing precise outcomes by means of brain organizations.

Brain networks are a type of model that mimics the neuron model in the human brain, allowing calculations to conduct circles on data to review and limit examples, and make do with the forecasts it generates with each circle. "Apple's Face ID is an outstanding instance of long-term learning. You train the calculation by examining your face while setting up your phone. When you sign in with Face ID, for example, the True Depth camera captures a large number of data points that form a depth map of your face, and the phone's inbuilt brain motor performs the investigation to figure out whether it is you ".

Although the research is limited to face recognition using Siamese Network and OpenCV, the fundamental concept may be replicated for any modular quality. I used Siamese Convolution Neural Networks for my investigation, which can determine the distance between two similar objects.These organizations are likewise prepared effectively utilizing standard improvement strategies on examined matches and furthermore present a cutthroat methodology that relies upon no particular space information which tackle the profound learning procedures. To extend the information in Siamese convolution brain organizations, I have expected to initially gain proficiency with a brain network with single secret layer and afterward numerous secret layers.The Multi-Layer Perceptron has been prepared using back engendering calculations, and then a Matrix type of back proliferation calculation has been constructed, which is leveraging several GPUs to perform more effectively. This investigation has shown that GPUs are capable of managing illustrations as well as improving processing time for certain calculations when used correctly.

Following the preparation of the photo dataset, the organization is presented with new image(s) to determine whether or not the organization perceives the face in the pictures.Alongside the face location and face acknowledgment, we have inferred a similar prepared organization to identify and perceive different items. By also picking the indicator for the organization to prepare and afterward forcing specific imperatives on the definition, this exploration has accomplished most extreme precision in facial acknowledgment, object acknowledgment and text recognition. Note that I don't propose to become familiar with the measurement so my strategies are focusing predominantly on the elements of profound learning. By and large, the thought is to initially become familiar with the great portrayals of brain organizations and afterward utilize the highlights to accomplish face identification and acknowledgment, object discovery and acknowledgment and text location with no retraining.

One of Artificial Intelligence's main goals is to enable computers to grasp the real world and give them the ability to assist or communicate with people. The goal of PC vision research is to understand and address the complicated visual reality. In most cases, individuals communicate with one another using standard language. The study of regular language pro-

cessing is important for human-computer communication, but it is also necessary to achieve "a clear" goal. The study of associations between images and texts is still in its early stages. Early work included generating single words or set sentences from photos [6, 18]. Picture inscribing refers to the creation of a whiz sentence depiction based on a central issue in order to achieve "a clear" goal.Connections between images Furthermore, text analysis is a growing topic of study. Early work included generating single words or set sentences from photos [6, 18]. Picture inscribing refers to the creation of a whiz sentence depiction based on a central issue in order to achieve "a clear" goal. The visual setting knowledge in PC vision and the sentence age in ordinary language handling are both included in programmed subtitling. Figure 1.1 shows an example of article acknowledgment, complete picture inscribing, and extensive subtitling (to differentiate notable locations and establish local portrayals). This idea seeks for ways to make visual inscriptions that are exact and unambiguous. The study of picture inscription is extremely important for obtaining sight and sound/scene, and it also has prospective uses that might aid social networks. It can, for example, aid visibly impaired people in obtaining their ambient components. The clever investigation, when combined with the discourse mix framework, may transform the visual world into a palpable experience. A sentence-level picture association is another possible use. It would be preferable if the image stage, such as Instagram or Google Photos, could combine our photos with semantic narrative rather than relying on the time stamp. These platforms might also provide clients with picture diaries. For these purposes, picture subtitling is a common method. Current research has revealed that the encoder-decoder structure is at the heart of many successful subtitle producers. Late developments in machine interpretation have energised these brain network strategies. A CNN encodes an image into a setting vector in this manner. The decoder, a Recurrent Neural Network (RNN), then translates the unique situation vector into a series of words. Many modifications install the consideration component and enable learning techniques based on this CNN-RNN structure. In light of the CNN-RNN system, we encourage the development of innovative modules.

1.1 Literature Review / Survey

Many models for picture subtitle undertakings have been planned after profound learning became well known. The standard calculations utilized CNN as an encoder and LSTM as a decoder. Despite the fact that models of picture inscription task are persistently different, CNN is, most assuredly, the best strategy to remove spatial data from a picture. In addition, LSTM gives promising outcomes for handling successive information. Due to the memory cell, the organization can create a current example in light of past examples. The blend of the use of encoder and decoder is, then again, very different in the writing. It can generally be partitioned

into six classes. Right off the bat, in Multi-modular organizations, two encoders are utilized to extricate both picture and text highlights. From that point forward, one decoder is used to deliver inscriptions. Besides, Semantic organizations centre not just around the order of the items on a picture yet additionally on the characteristics of the items. Thirdly, Attention models do just not use picture highlights from CNN's last layer for once. While creating words, the models likewise realize where and what to zero in on a picture highlights. Fourth, the quantity of various items and words in depictions in datasets is negligible contrasted with this present reality. To stay away from this impediment, Zero-shot learning is utilized. At long last, average picture subtitle strategies are based on directed learning. Support learning and Unsupervised learning have moreover turn out to be very well known as of late. In the accompanying areas, These six classes will be made sense of with related papers.[2]

1.1.1 Multi-modal networks

Multi-modular organizations are one of the principal networks which use profound learning calculations. As referenced above, picture highlights guide and text highlights are utilized in the picture subtitle. Also, a learning model ought to comprehend the connection between picture and text highlights. Nonetheless, these highlights are not characterized in something similar space. In this way, Multi-modular organizations planned them into a similar space. One of the main paper which utilizes profound learning is . In , comparing portrayals of a picture are inspected in an equal way. As opposed to the past model, it utilizes a convolution brain organization to extricate picture highlights from the picture. Besides, it likewise uses a feed-forward brain organization to plan words in the datasets for word inserting. There are two distinct models in the paper. In the main model, Multimodal Log-Bilinear Models are utilized for planning words and picture highlights to the same space. While creating the current word, the summation of weighted implanting words and weighted inserted picture highlights are shipped off a softmax layer. It is likewise whenever that backpropagation first assists the organization with uniting weight to their optimal qualities. The subsequent technique utilizes Factored 3-way Log-Bilinear, which analyses picture highlights like a door to plan picture elements to words. Then, at that point, it maps the outcome to the softmax layer. In any case, Multimodal Log-Bilinear isn't the main technique to plan text highlights. additionally encodes picture highlights utilizing CNN, yet in this time, it uses RNN for text highlights. Then, these picture highlights and text highlights are planned to multi-modular space utilizing a completely associated brain organization. Also, both picture also, word inserted vectors are joined. Then, another vector installing is made in a multi-modular space. Furthermore, likewise presents another decoder strategy which is called Structure-Content Neural Language Model. The decoder figures out how to create grammar right words utilizing a given substance inserting vector, word-explicit structure, and past words in this technique.

As referenced above, picture highlights are utilized to produce a sentence. In any case, text elements can likewise be utilized to reproduce picture highlights. For example, centres around a bi-directional planning between a sentence and a picture. The thought comes from the representation of a picture by perusing a sentence. Be that as it may, to make this cycle simpler, they attempt to re-create picture highlights rather than the entire picture. During the preparation, the model endeavours to reproduce given picture highlights and produce sentences all the while. Dormant factors become an extension between these two undertakings. While re-producing picture highlights, past words are encoded to inactive factors utilizing RNN. Then, at that point, idle factors are changed into picture highlights. Furthermore, to produce sentences, picture highlights are encoded to dormant factors. Then, the current word is anticipated in light of past words and inert factors. Observing an appropriate sentence doesn't continuously have to produce a sentence. A comparability score can likewise be utilized. In , rather than delivering a clever depiction, it positions picture sentence pair and carries to the front the sentence with the most elevated score. To find the most important picture sentence pair, all picture sentence matches are relegated a score which is the internal result of visual inserting and text implanting. Plus, picture highlights are gotten utilizing RCNN, and sections of visual embedding's and text implanting are changed over to the multi-modular space utilizing a brain organization. Hence, the model is prepared for the two parts of visual and text matches and all pieces and sentences. As opposed to , a few papers mean to produce more clever sentences. For instance, J.Mao planned another design which is called m-RNN in . In this organization, two inserting layers can change over the one-hot-encoded word vector to message installing. Then, message highlights are introduced to RNN. Notwithstanding text highlights, picture highlights are additionally introduced to the RNN to keep away from neglecting picture portrayal. Then, at that point, text elements of past words and picture highlights are planned to a Multi-modular space utilizing a brain organization. The softmax layer is likewise applied to anticipate the current word.

1.1.2 Semantic Networks

Ascribes are fundamental in the AI region. The shading, shape, surface of an article are various sorts of characteristics. CNN separates just a theoretical portrayal of a given picture. Hence, despite the fact that a quality's data is acquired at CNN's low-level layer, it very well may be lost at CNN's undeniable level layer. Semantic organizations additionally intend to take advantage of all subtleties of articles on a picture utilizing their qualities. Properties can be mastered utilizing different organizations that emphasis on ascribes as it were. For instance, train extra a characteristics identifier utilizing Multiple Instance Learning, which is a pitifully regulated technique, to recognize credits. As a decoder, there are three sorts of LSTM utilization in . To introduce these characteristics to the organization, they plan

LSTM-A. In addition, five sorts of LSTM-An are referenced in the paper. The distinctions are the request for introducing picture elements and characteristics to the organization. Also, produces its subtitles with more than one credits indicators utilizing a big picture perspective and a granular perspective. The contrast between these methodologies is made sense of as following: While the hierarchical technique utilizes visual picture portrayal and afterward begins creating words, the base up process initially produces words which portray picture then blend them likewise consolidates these two methodologies by working out semantic consideration. Additionally, after picture highlights are gotten, they are introduced to the organization just a single time. Then, at that point, distinguished ascribes are utilized to figure the information consideration score and result consideration score. These two consideration scores are used to make another information vector to RNN and another result vector for the Softmax layer. A few papers, then again, motivates from the human mind to make more book sentences., for example, asserts that normal LSTM can't actually make novel subtitles from portrayals of pictures. The calculations in the writing typically recognize ascribes prior to identifying objects in the picture. Like human cerebrum, distinguishes the items first and afterward credits. For this reason, to prepare network for properties and rest of the sentences, qualities and rest of the sentences are recognized involving Stanford electorate parser which characterizes words in sentences as a thing, thing phrase, and so forth Furthermore, there are two kinds of LSTM; Skel-LSTM and Attr-LSTM. Picture highlights are first shipped off Skel-LSTM. The skeleton of the subtitle is assembled. Then, Attr-LSTM is prepared to give credits for skeleton sentences. As opposed to trains another organization to get ascribes. While preparing, K familiar words in the portrayal are identified for a given picture, and they are viewed as semantic marks. Then, at that point, these semantic ideas are prepared like multi-label characterization. A short time later, distinguished semantic elements are joined with word embedding vectors and a secret layer of LSTM for each progression while producing yield words for a given picture.

1.1.3 Reinforcement Networks

Conventional picture subtitle undertakings utilize directed learning. In any case, after ongoing upgrades in Reinforcement learning, new papers have additionally begun to be utilized Reinforcement calculations in picture subtitle errands. In this learning system, the model makes moves in a climate. In light of these activities, the model loads are refreshed by remuneration esteem rather than a misfortune, for example, in directed learning. The design is to gather awards however much as could be expected. In the Reinforcement learning part of picture inscription, the pictures are our current circumstance and produced words from the model are the activities. Many models train a worth organization as opposed to utilizing a prize capacity. is one of the models which uses a worth organization. In these undertakings,

there are two networks which are an arrangement organization and a worth organization. Like directed strategies, the Policy network prescribes a word to the model for each progression utilizing LSTM. Later that, the worth organization attempts to track down the word with the most elevated score in the long haul. To do that, it links installing picture elements and message elements and presents them to multi-facet perceptron to acquire a prize worth. The strategy network then, at that point, refreshes their loads in view of this award. Another model is on an actor critic model for picture inscription errands. The Policy organization (Actor) delivers another word at each progression while the Value organization (pundit) censures the current created word. As opposed to utilizing MLP, involves the Cider score work as a prize capacity. Be that as it may, Cider can't give a solid assessment score by contrasting just a word and a sentence. Hence, the Value network expects that the Policy organization would continue to create words in view of current and past words. The outcome inscription is analysed with depictions. This interaction is additionally characterized as an expected potential compensation score for the current word. Then, Similar to , the Policy network is prepared for the current word utilizing this potential compensation score. expects to create more regular, different, and semantically near the ground truth. In , they guarantee that since customary strategies train their model in light of amplifying probability assessment, these techniques can't make unmistakable inscriptions from portrayals in datasets. Accordingly, consolidates GAN and Reinforcement Learning to keep away from this issue. It trains two distinct organizations; Generator and Discriminator. Generator manages creating sentences in view of a given picture. For variety, an irregular vector is likewise expected as info. Words are expressed as activity and are chosen in view of the approach of the organization. Notwithstanding, a lot of words can't be utilized to work out a prize. Hence, the Expected potential compensation is determined for each word by letting create the remainder of the sentence at n times for the current word. From that point forward, the normal compensation of these sentences is determined, and Generator's boundaries are refreshed. Additionally, the discriminator is utilized as a prize capacity. It establishes that a given sentence is from Generator or a sentence from the dataset. Eventually, Generator what's more, Discriminator are prepared in view of various goals. While the Generator attempts to hoodwink Discriminator about the produced sentence made by a human, Discriminator attempts to comprehend the distinction between an inscription and a portrayal in the dataset.

1.1.4 Unsupervised Networks

Unsupervised learning is another learning mechanism. In contrast to Reinforcement learning and supervised learning, the purpose is to teach a model to generate a proper caption without using any image-description pair. is one of the first algorithms. The model transforms both texts and images to joint embedded space using RNN and CNN as an encoder. After extracting

image features using CNN, image features are mapped to joint space using a multi-layer perception layer. To train MLP, a discriminator learns to distinguish joint embedded image features and visual concepts. also aims to create connections between objects in the image and related objects in the image, such as 'a TV' and 'a TV table'. In this way, the model generates more novel sentences than the description in the dataset. Additionally, the RNN encoder learns to encode given text. In producing captions, the RNN decoder is very similar to the RNN encoder. However, in the former, text embedded vectors and image embedded vectors are received as inputs.

1.2 Motivation

In web development, it's good practice to provide a description for any image that appears on the page so that an image can be read or heard as opposed to just seen. This makes web content accessible. Automatic Captioning can help, make Google Image Search as good as Google Search, as then every image could be first converted into a caption and then search can be performed based on the caption. CCTV cameras (Closed-circuit television cameras) are everywhere today, but along with viewing the world, if we can also generate relevant captions, then we can raise alarms as soon as there is some malicious activity going on somewhere. This could probably help reduce some crime and/or accidents.

1.3 Problem Statement

we are trying to achieve the real time facial emotion detection with back ground Dependent and Accordingly Auto Captioning the expression along with object detection.

1.4 Objectives

- Multiple person Facial Emotion Detection in a single frame.
- Along with Emotion of Person auto captioning the present situation.
- Object detection in the frame and auto captioning.
- Implement the above steps in the Real time with WEB based application and with ML Architecture and functioning.

Chapter 2

REQUIREMENT ANALYSIS

This chapter gives a brief description about Functional Requirement, Non Functional Requirement, Hardware Requirement, Software Requirement details of our system

2.1 System Model

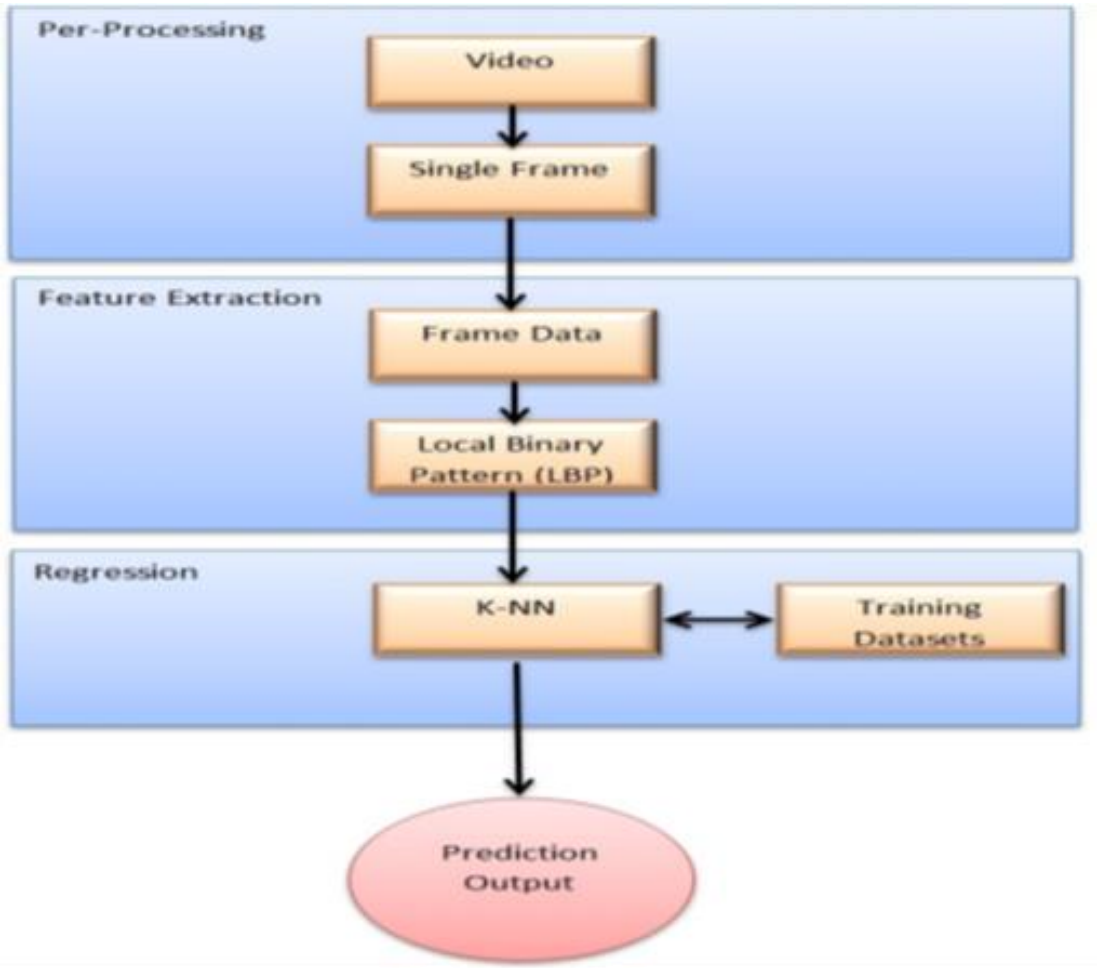


Figure 2.1: Detailed design of the system model for the project

2.2 Functional Requirements

The functionality of a system or one of its subsystems is defined in a functional requirement document. It also relies on the software, the expected users, and the machine on which the software is installed.

- Multiple person Facial Emotion Detection in a single frame.
- Along with Emotion of Person auto captioning the present situation.
- Object detection in the frame and auto captioning.
- Implement the above steps in the Real time with WEB based application and with ML Architecture and functioning.

2.3 Non Functional Requirements

Functional requirements are to be provided by the application. The functional requirements for the applications:

- User shall be able to simulate the web app on their own.
- User can view their name, age, gender and even nationality on their own in the real time.
- User shall able to interact with the preferred algorithms behind the website.
- person with programming skills shall be able to update the existing models
- programmer shall be able to edit or delete the modules present in them, for real time as well as static images.

2.4 Software Requirements

- Python (any version):
  We choose python because the python language is one of the most accessible programming languages available because it has simplified syntax and not complicated, which gives more emphasis on natural language. Due to its ease of learning and usage, python codes can be easily written and executed much faster than other programming languages. Python is a general-purpose programming language, so it can be used for many things. Python is used for web development, AI, machine learning, operating systems, mobile application development, and video games.. The community hosts conferences and meetups, collaborates on code, and much more. Python is developed under an OSI-approved open source license, making it freely usable and distributable, even for commercial use. Python's license is administered by the Python Software Foundation.

- Jupyter Notebook:
  We use jupyter notebook because Jupyter is a free

  open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document. The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser. The Jupyter Notebook App can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet

2.5 Hardware Requirements

- System: Intel core i5
- Operating System: Windows 8th Gen
- Hard disk: 40GB
- Monitor: 15VGA Color
- Ram: 8GB

Chapter 3

SYSTEM DESIGN

The process of establishing system aspects such as modules, architecture, components and their interfaces, and data for a system based on specified requirements is known as systems design. It is the process of identifying, creating, and designing systems that meet a company's or organization's specific objectives and expectations.The goal of the System Design process is to give enough precise data and information about the system and its components to allow for implementation in accordance with the architectural entities established in the system architecture's models and views.This chapter will gives a brief description about System Design which gives brief about Architecture Design, Data Design, User Interface Design of our project

3.1 Architecture Design

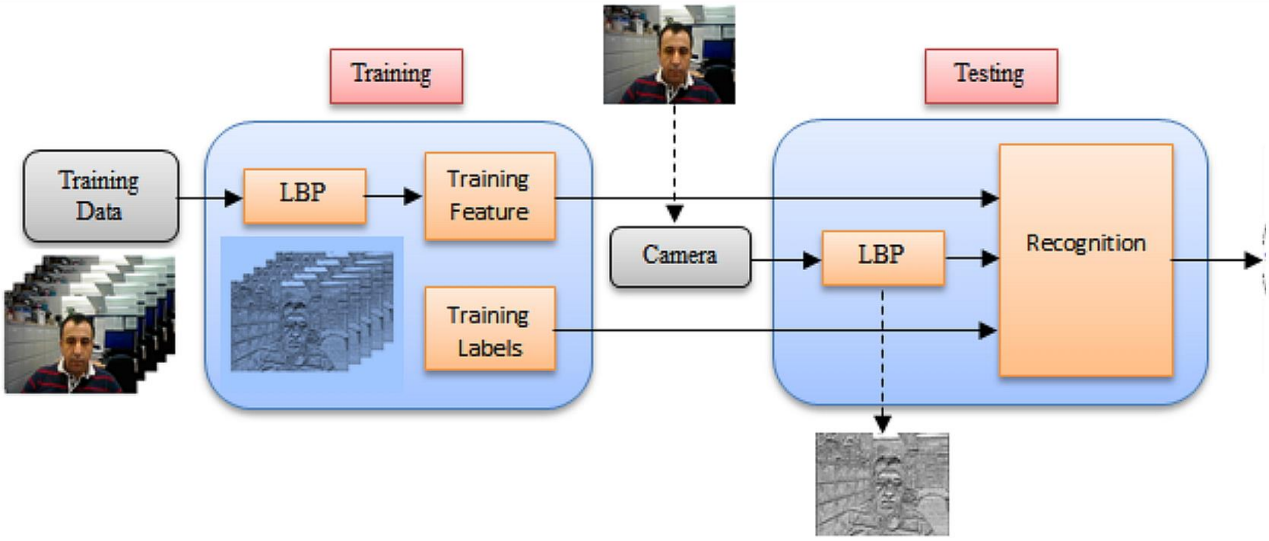


Figure 3.1: Real-time emotion detection system from facial expression on embedded devices.

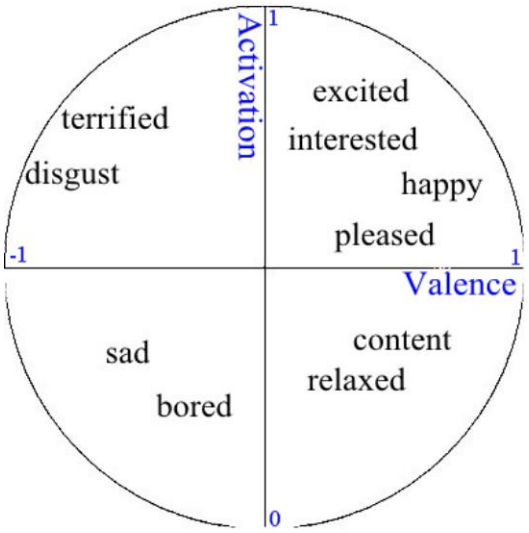


Figure 3.2: activation with valance diagram

3.2 Methodology

3.2.1 Deep learning and neural networks

Not at all like human mind which is incredibly great at sorting out the things the eyes shows us, a brain network utilizes absolutely an alternate way to deal with learn and see this present reality things. A brain Network moves toward the issue of visual example or picture handling another way. The thought behind the working of brain network is to take gigantic data set of pictures, which are likewise, called preparing models and grow a framework which can gain from these preparation models. A brain network step by step turns out to be more productive when it takes care of itself more with such preparation models along these lines working on its exactness in anticipating the right outcomes without human intercession. Motivation behind the Artificial Neural Network is the way organic brain networks in the human cycle data. Neuron is the essential and fundamental computational part of the human cerebrum. Moreover, the fundamental computational unit in a counterfeit brain network is the neuron which is additionally known by the name of hub or a unit. It gets an information which is related with some weight, from an outer source or another unit (hub) and works out or figures a result. The weight which has been related to the sources of info has been doled out to them in view of the relative significance to different data sources . There is a capacity in the unit called Activation Function, which is applied to the weighted amount of the multitude of contributions alongside one more information called inclination, additionally connected with a weight, to process a result.

The enactment work is non straight and consequently, is applied to the result to carry

non linearity to the result since every one of this present reality information is non direct and the brain network is tied in with learning the non straight portrayal of genuine world. The fake neuron is likewise called Perceptrons. There are various variations of Artificial Neural Networks and these variations further have their own sorts or classes.

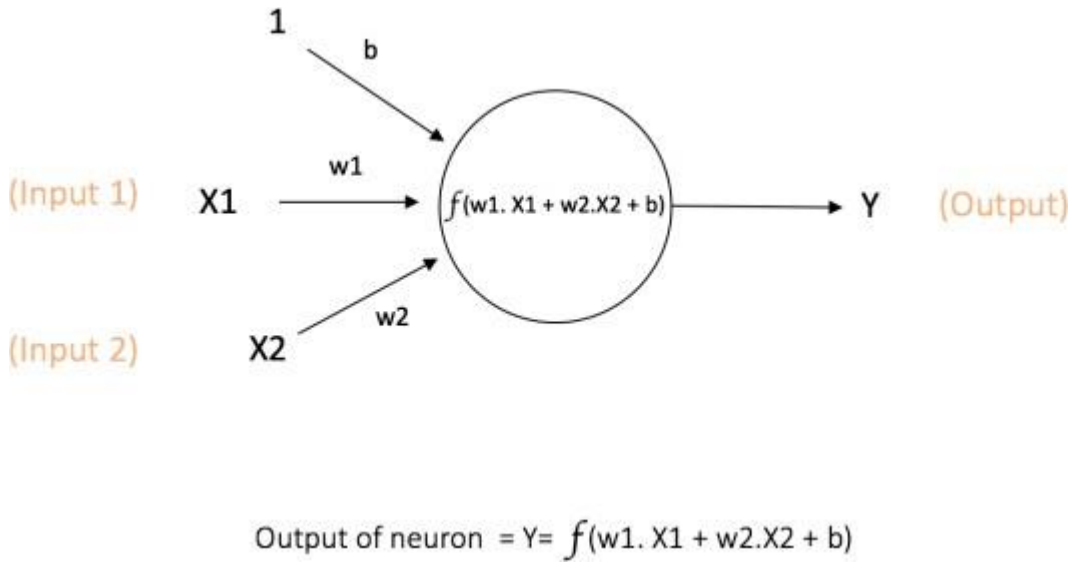


Figure 3.3: A Single Neuron

The most commonly used neural networks are Feed Forward Neural Network, Single Layer Perceptron, Multi-Layer Perceptron (MLP), Convolution Neural Network (CNN), and Recurrent Neural Networks (RNN). And the most commonly used Activation Functions which are found more often in practice are Sigmoid, tanh, ReLU, Leaky ReLU. Every activation function takes a single number and performs a certain fixed mathematical operation on it ..

- Sigmoid: takes a real-valued input and squashes it to range between 0 and 1 (x) = 1 / (1 + exp(x))

- tanh: takes a real-valued input and squashes it to the range [-1, 1] Tanh(x) = 2 (2x) 1

- ReLU: ReLU stands for Rectified Linear Unit. It takes a real-valued input and thresholds it at zero (replaces negative values with zero) f(x) = max(0, x)

The below figures . show each of the above activation functions.

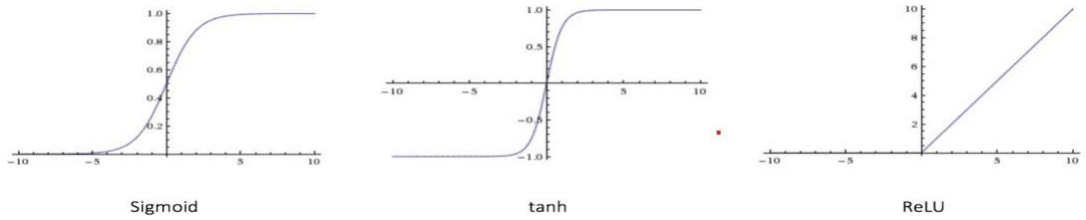


Figure 3.4: Different activation function

3.2.2 Types of Neural Networks

3.2.2.1 Feed Forward Neural Networks

T he first and the simplest form of the artificial neural network conceived was the Feed Forward Neural Network. It has various layers which are arranged and are composed of multiple nodes (neurons). Neurons from neighboring layers are interconnected and all the links or associations have weights assigned to them.
The below figure shows the instance of Feed Forward Neural Network.

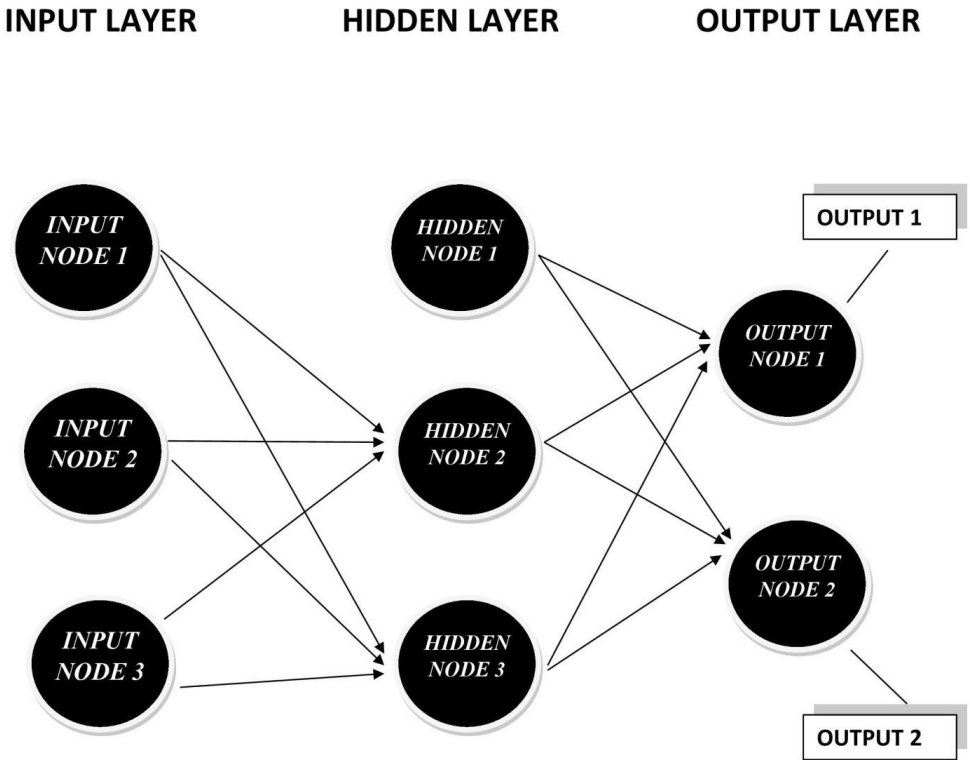


Figure 3.5: An example of Feed Forward Neural Networ

A Feed forward neural network consists of three types of nodes

- Input Nodes – The Input nodes provide information from the outside world to the network and are together referred to as the "Input Layer". No computation is performed in any of the Input nodes – they just pass on the information to the hidden nodes .
- Hidden Nodes – The Hidden nodes have no direct connection with the outside world (hence the name "hidden"). They perform computations and transfer information from the input nodes to the output nodes. A collection of hidden nodes forms a "Hidden Layer". While a feed forward network will only have a single input layer and a single output layer, it can have zero or multiple Hidden Layers .
- Output Nodes – The Output nodes are collectively referred to as the "Output Layer" and are responsible for computations and transferring information from the network to the outside world .

In the feed forward neural networks, the information moves only in one direction unlike recurrent neural networks which forms a loop. Feed forward neural network moves the information from input layer though the hidden layer to the output layer. Two types of feed forward neural networks are "Single Layer Perceptron" and "Multi Layer Perceptron".

3.2.2.2 Multi Layer Perceptrons (MLP).

A multi layer Perceptron is a neural network with multiple input layers connected as a one way directed graph to the output layer, which generates outputs from a set of inputs. Each node in this network has the non linear activation function except the input layer. Multi Layer neural network has the best practice in the field of supervised learning as well as research into parallel and distributed processing involving the supplication of speech recognition, image recognition and machine translation. MLP uses back propagation algorithm as a supervised learning approach. As there are multiple layers of neurons present in this neural network, it is basically a deep learning system.

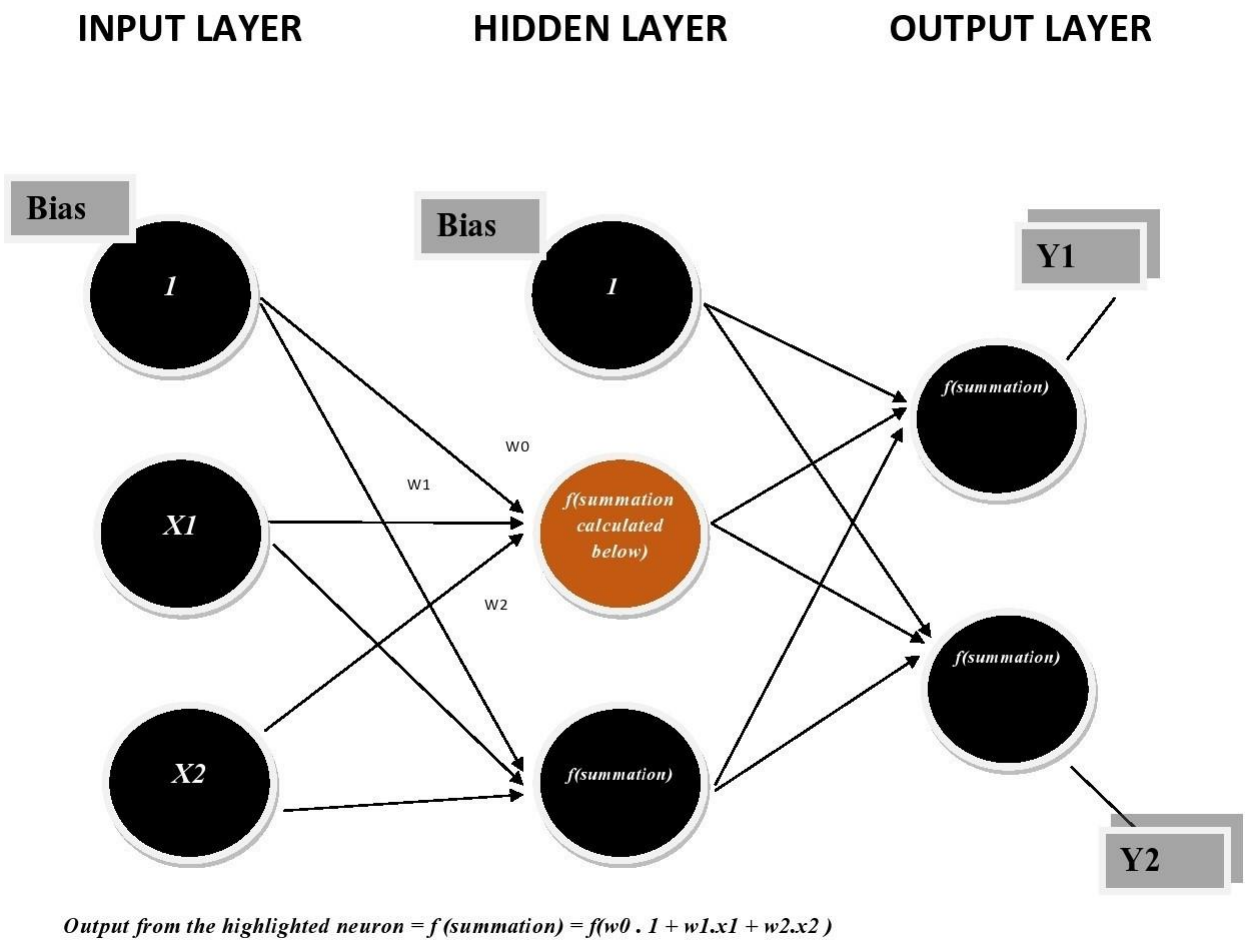


Figure 3.6: Multi Layer Perceptron having one hidden layer.

So far, I've studied about the neural networks in which input of the one layer comes from the output of the previous layer. The information flow is one way i.e. the information is only fed forward and never fed backwards. These types of networks are called feed forward neural networks which mean there are no loops in the network. There are other models of neural networks which have feedback loops in them. These models are called Recurrent Neural networks.

3.2.3 The Back Propagation Algorithm

"Back propagation is a method used in artificial neural networks to calculate a gradient that is needed in the calculation of the weights to be used in the network". Deep Neural networks are trained using back propagation by distributing errors throughout the network layers backwards. The back propagation algorithm was proposed in 1986 by Rumelhart, Hinton and Williams for settling weights and hence for the training for Multi Layer Perceptron (MLT).

Derivation of Back propagation starts with computing the output layer which is the only one where desired outputs are available but the outputs of the intermediate layers are unavailable. Back propagation revolves around four fundamental equations and together these four equations provide us the insights to calculate both the errors and the gradient of the cost function. Back propagation equations, at a level, goes so deep that understanding them well requires considerable time and patience. All four equations are the consequences of the chain rule from multivariable calculus. "Equation 1 (BP1): which gives an expression for the output error, L. To prove this equation, recall that by definition":

$$\delta_j^L = \frac{\partial C}{\partial z_j^L}.$$

"Applying the chain rule, we can re-express the partial derivative above in terms of partial derivatives with respect to the output activations",

$$\delta_j^L = \sum_k \frac{\partial C}{\partial a_k^L} \frac{\partial a_k^L}{\partial z_j^L},$$

"where the sum is over all neurons k in the output layer. Of course, the output activation aLk of the kth neuron depends only on the weighted input zLj for the jth neuron when k=j. And so aLk/zLj vanishes when kj. As a result we can simplify the previous equation to"

$$\delta_j^L = \frac{\partial C}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L}.$$

"Recalling that aLj=(zLj) the second term on the right can be written as (zLj), and the equation becomes"

$$\delta_j^L = \frac{\partial C}{\partial a_j^L} \sigma'(z_j^L),$$

"Next, we'll prove (BP2), which gives an equation for the error l in terms of the error in the next layer, l+1. To do this, we want to rewrite lj=C/zlj in terms of l+1k=C/zl+1k. We can do this using the chain rule" .,

$$\begin{aligned} \delta_j^l &= \frac{\partial C}{\partial z_j^l} \\ &= \sum_k \frac{\partial C}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} \\ &= \sum_k \frac{\partial z_k^{l+1}}{\partial z_j^l} \delta_k^{l+1}, \end{aligned}$$

"Where in the last line we have interchanged the two terms on the right-hand side, and substituted the definition of l+1k. To evaluate the first term on the last line, note that"

$$z_k^{l+1} = \sum_j w_{kj}^{l+1} a_j^l + b_k^{l+1} = \sum_j w_{kj}^{l+1} \sigma(z_j^l) + b_k^{l+1}.$$

"Differentiating, we obtain",

$$\frac{\partial z_k^{l+1}}{\partial z_j^l} = w_{kj}^{l+1}\sigma'(z_j^l).$$

"Substituting back into (2), we get",

$$\delta_j^l = \sum_k w_{kj}^{l+1}\delta_k^{l+1}\sigma'(z_j^l).$$

"This is just (BP2) written in component form". "Let's explicitly write this out in the form of an algorithm,

- Input x: Set the corresponding activation a1 for the input layer .
- Feed forward: For each l=2,3,. . . ,L compute zl=wlal1+bl and al=(zl) ..
- Output error L: Compute the vector L=aC(zL) .
- Back propagate the error: For each l=L1,L2„2 compute l=((wl+1)Tl+1)(zl) ..
- Output: Gradient of the cost function is given by Cwljk=al1klj AND Cblj=lj" .

Analyzing the algorithm and the equations, one can see the reason behind its name back propagation. Starting from the final layer, the error vector is determined backward. To understand how the cost varies with earlier weights and biases we need to repeatedly apply the chain rule, working backward through the layers to obtain usable expressions.

3.2.4 Matrix Form of Back Propagation using CUDA

To utilize the GPU performance to the fullest, matrix form of batch mode back propagation algorithm is implemented by introducing input neurons, output neurons and hidden neurons into the matrix. The implementation of the algorithm includes CUDA Basic Linear Algebra Subroutines (cuBLAS) library to perform vector and various matrix functioning along with CUDA kernel. It enables various GPUs at the same time with the same neural network structure and weight parameters. The training examples are distributed to various GPUs and each GPU in return calculates the local training error and the gradient at each layer which is sent back to the first GPU to calculate the summations which are further transferred back to update the weights until the training goal is achieved. This technique has achieved higher efficiency than the conventional GPU implementation of the Back Propagation algorithm .

Having a neural network with multi layers, suppose the total number layers are L such that the first layer is the input layer and the output layer is Lth layer with hidden layers through 2 to L-1. And the number of neurons in lth layer be Nl. Given the inputs x =[x1, x2,..., xn]T, where n = N1 is the number of input neuron. Feed forward in neural networks is used to computer outputs y = [y1, y2, , ym]T where m is the number of output neurons . Let s be the number of training samples and in order to take the bias of hidden neurons, one can add a fake neuron with value to be 1, into the input layer and each hidden layer . Performing training of back propagation in matrix Form requires the input matrix X with dimensions of s x (n+1) which is initialized to store all the input data in training samples with the neuron which is fake . The values of the elements in (n+1)th column of matrix X are one .

The weight matrix W1 at the lth layer has the dimensions of (Nl -1 +1) x Nl with all the elements in Wl are random values representing the guess . Let the matrix Zl be with be the dimensions of s x (Nl-1 + 1) representing the value of neurons at the lth layer. The elements in (Nl + 1) th column of Zl also representing the bias at the lth layer . The output matrix Y is of the dimension's s x m where each row of Y represents the output of the corresponding row of one trained sample in the input matrix X.

The first step towards this is to feed forward the input data to the output neurons where the value of each layer is calculated as

$$\boldsymbol{Z}^{1}_{s\times N_1} = \boldsymbol{X}_{s\times n}$$
$$\boldsymbol{Z}^{l}_{s\times N_l} = \sigma(\boldsymbol{Z}^{l-1}_{s\times (N_{l\ 1}+1)}\boldsymbol{W}^{l}_{(N_{l\ 1}+1)\times N_l}), l = 2, 3, \ldots, L-1$$

Where Zls X Nl demonstrates only the elements in the first column which are updated while the elements in (Nl x 1)th column of Zl remains to be unchanged . () is the activation function of the hidden neurons which is also known as the Sigmoid activation function, which applies to each of the product of Z(l-1) and Wl. The output matrix is pulled out as

$$\boldsymbol{Y}_{s\times m} = \boldsymbol{Z}^{L-1}_{s\times (N_{L\ 1}+1}\boldsymbol{W}^{L-1}_{(N_{L1\ 1}+1)\times m}$$

The next step is to calculate the training error. A matrix D with the dimensions of s x m is made to store all the outputs in training samples i.e. s as desired output data. The error is calculated by the matrix form of the difference between NN outputs and the desired outputs as

$$\boldsymbol{E} = \frac{1}{2} sum((\boldsymbol{Y}_{s\times m} - \boldsymbol{D}_{s\times m}) \odot (\boldsymbol{Y}_{s\times m} - \boldsymbol{D}_{s\times m}))$$

Where donates product of two matrices and sum () is the summation of all elements in the matrix .

The third step is to calculate the gradient at each layer and the total gradient is the summation of the entire training samples gradient. Let Gl be the s x p matrix which represents the gradient at the lth hidden layer and Gl be the s x m matrix that represents the gradients at the output layer which are computed as

$$\boldsymbol{G}^{l}_{s\times N_l} = \boldsymbol{Z}^{l}_{s\times N_l} \odot (\boldsymbol{U}^{l}_{s\times N_l} - \boldsymbol{Z}^{l}_{s\times N_l}), l = 2, 3, \ldots, L-1$$
$$\boldsymbol{G}^{L}_{s\times m} = \boldsymbol{D}_{s\times m} - \boldsymbol{Y}_{s\times m}$$

Where UL is the s x NL unity matrix with all the elements as 1.

The last step is to back propagate the gradient from (l+1)th layer to the lth layer. The weights between the input layer and the first hidden layer and the rest of the hidden layers can be calculated as

$$\begin{aligned}
&\Delta \boldsymbol{W}^{l}_{(N_{l\ 1}+1)\times N_l}\Big|_{\boldsymbol{w}^{l\ 1}=\boldsymbol{W}^{l\ 1}_{now}} = \\
&-\eta \cdot \{\boldsymbol{Z}^{T}_{(N_{l\ 1}+1)\times s}[\boldsymbol{G}^{l}_{s\times N_l}{}_{\ 1}\boldsymbol{W}^{l}_{N_{l\ 1}\times N_l} \odot \boldsymbol{G}^{l}_{s\times N_l}]\} \\
&+\alpha \cdot \begin{bmatrix} \boldsymbol{W}^{l}_{(N_{l\ 1}+1)\times N_l}\Big|_{\boldsymbol{w}^{l\ 1}=\boldsymbol{W}^{l\ 1}_{now}} \\ -\boldsymbol{W}^{l}_{(N_{l\ 1}+1)\times N_l}\Big|_{\boldsymbol{w}^{l\ 1}=\boldsymbol{W}^{l\ 1}_{old}} \end{bmatrix}, l = 2, 3, \ldots, L-1
\end{aligned}$$

Finally, the weights between the last hidden layer and the output layer can be calculated as

$$\Delta \boldsymbol{W}^{L}_{(N_{L\ 1}+1)\times N_L}\Big|_{\boldsymbol{w}^{L\ 1}=\boldsymbol{W}^{l\ 1}_{now}} = -\eta\cdot[\boldsymbol{Z}^{T}_{(N_{L\ 1}+1)\times N_L}\boldsymbol{G}^{L}_{s\times N_L}] + \alpha\cdot\begin{bmatrix} \boldsymbol{W}^{L}_{(N_{L\ 1}+1)\times N_L}\big|_{\boldsymbol{w}^{L\ 1}=\boldsymbol{W}^{l\ 1}_{now}} \\ -\boldsymbol{W}^{L}_{(N_{L\ 1}+1)\times N_L}\big|_{\boldsymbol{w}^{L\ 1}=\boldsymbol{W}^{l\ 1}_{old}} \end{bmatrix}$$

The implementation of GPU enables Back Propagation Algorithm is supported by cuBLAS library and the CUDA kernel where cuBLAS library is the CUDA implementation of the basic linear algebra operations which supports three basic operations which are vector, matrix vector and matrix-matrix operations. CUDA kernel which implements Sigmoid function is the function defined by the user that runs on the GPU and performs the operations of same type to different data in order to take the advantage of GPUs highly parallel stream processing . The initialization of CUDA kernel allocates the memory space and transfers the data from system memory to GPU memory and after the matrices are allocated on the GPU memory, they will be reserved on the GPU memory with updated values which were trained during the process. This technique does a very distinctive job in sending back the total error to the system memory once per iteration when all the training samples feed forwarded to the output layer instead of sending back the error of just one training sample as done in , which has minimized the overhead of transferring data between the GPU memory and system memory. Also it has taken advantage of the parallel stream processors on the GPU . A self activating technique of multiple GPU training is considered where the number of GPUs in the system is g . The total sets of training examples are distributed across GPUs and the input matrix X, the output matrix Y and the desired output matrix D are systematically scaled to contain nearly s/g rows of data as shown. Each GPU feed forwards the input data to the output neurons and computes the training errors and the gradient . The GPU 1 will accumulate the error and the gradient information from all other GPUs and computes the summation of the training error and the summation of the gradients respectively . The total of the gradients are sent to each GPU to update the weights while the training error is sent to the system memory after which the CPU determines if the training objective has been achieved or not .

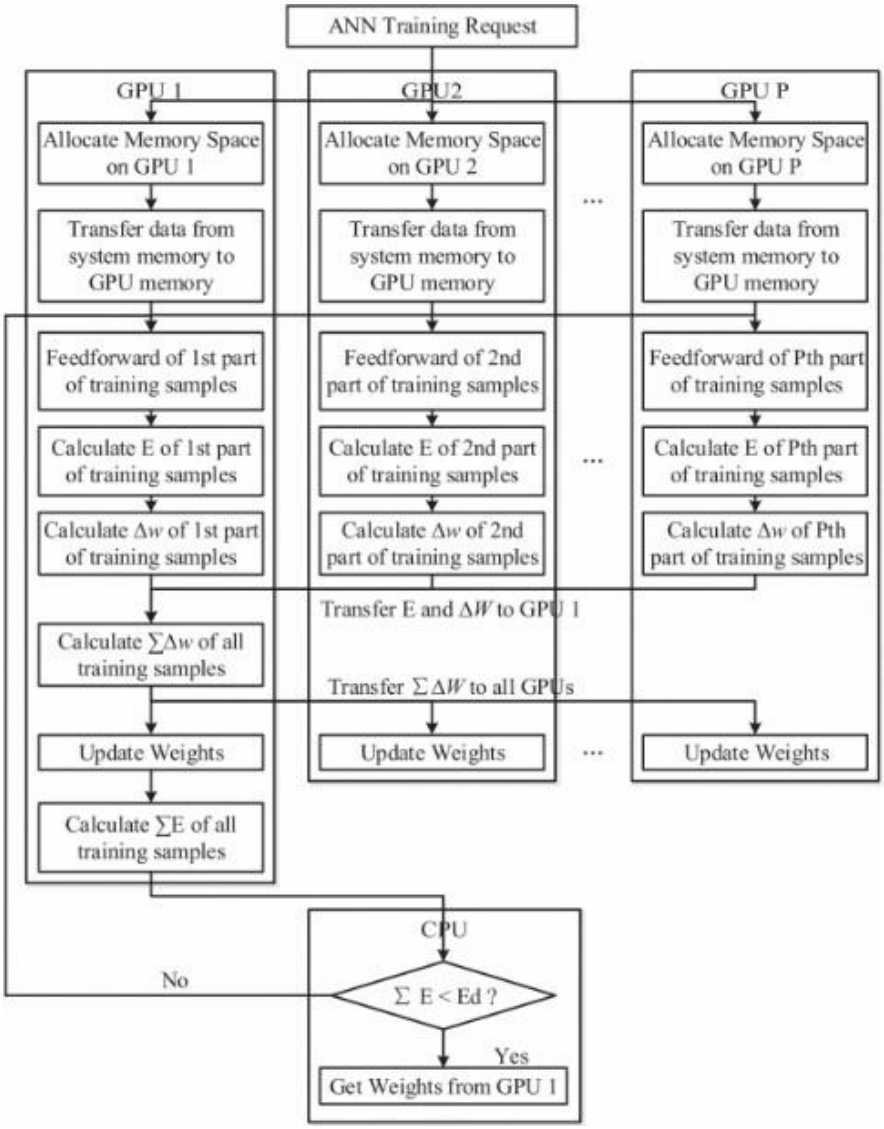


Figure 3.7: Framework of the parallel back propagation training on the multiple GPUs

3.2.5 Face Recognition using Deep Convolutional Neural Network.

This part of the research is fairly immature yet and is mainly based on the system known as Facenet. Facenet uses the Deep Convolutional Neural Network. In the experiment above, where matrix form of back propagation is introduced, I have trained the CNN using the Stochastic Gradient Descend (SGD). Even in the standard back propagation algorithm the same training technique was used . The model is initialized from random, similar to , and trained on the CPU cluster for more than 2 hours, having the dataset of 1000 images. Given the model details , and treating the model as the black box , the most important part of this approach lies in the end to end learning of the whole system . To this end employ the triplet loss that directly reflects what want to achieve in face verification, recognition and clustering.

Has not done any comparison to other losses but the motivation is that the loss from encourages that all faces from one identity to be projected onto the single point. The triplet loss enforces a margin between each pair of faces which allows the face of one identity to live on a manifold meanwhile the distance is enforced to discriminate with other identities. Triples Loss is motivated in in the context of nearest neighbour classification. Here it is ensured that the image of a specific person comes closer to all other images of the same person than it is to any image of any other person.

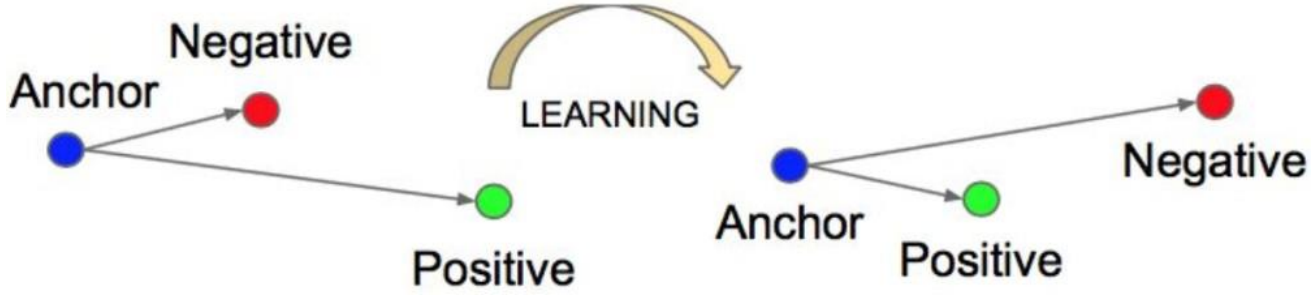


Figure 3.8: Triplet Loss .

Therefore the loss (L) is:

$$\sum_{i}^{N} \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+ \qquad \alpha = 0.2$$

Many would easily satisfy the above constraint, so it will be a waste to look at these during the training as it wouldn't contribute to adjusting parameters instead would only slow down the convergence. It is important to pick hard triplet that are active so that they can contribute to improving the model.

3.2.6 Face Recognition using deep learning and openCV

This part of the research is mainly based on the open source computer vision library called OpenCV. This part of the research is implementing OpenCV to perform face recognition. To build the system, I performed face detection at first and then extracted facial embeddings from each face using deep learning and trained a face recognition model on those embeddings and then at last came up with the system that finally recognize faces in both images and videos streams. While it was possible and easy to recognize faces using the OpenCV library, the fact that OpenCV itself is not responsible for identifying faces is a vibration. Along with OpenCV and deep learning, I have used another library called scikit-learn. This library is used in detecting faces, computing 128-d face embeddings to quantify a face, train a support vector machine (SVM) on the top of the embeddings to recognize faces in images and video streams.

3.2.6.1 How it works

Deep learning is applied in two simple steps in order to build the OpenCV face recognition system pipeline. The first step is to apply the face detection which will detect the presence of the face in an image or video stream but will not identify it and the second step is to extract the 128d feature vectors that quantify each face in an image or video stream. These vectors are also known as embeddings. One can perform fast and accurate face detection with OpenCV along with the pre trained deep learning face detector model that is shipped with the library.

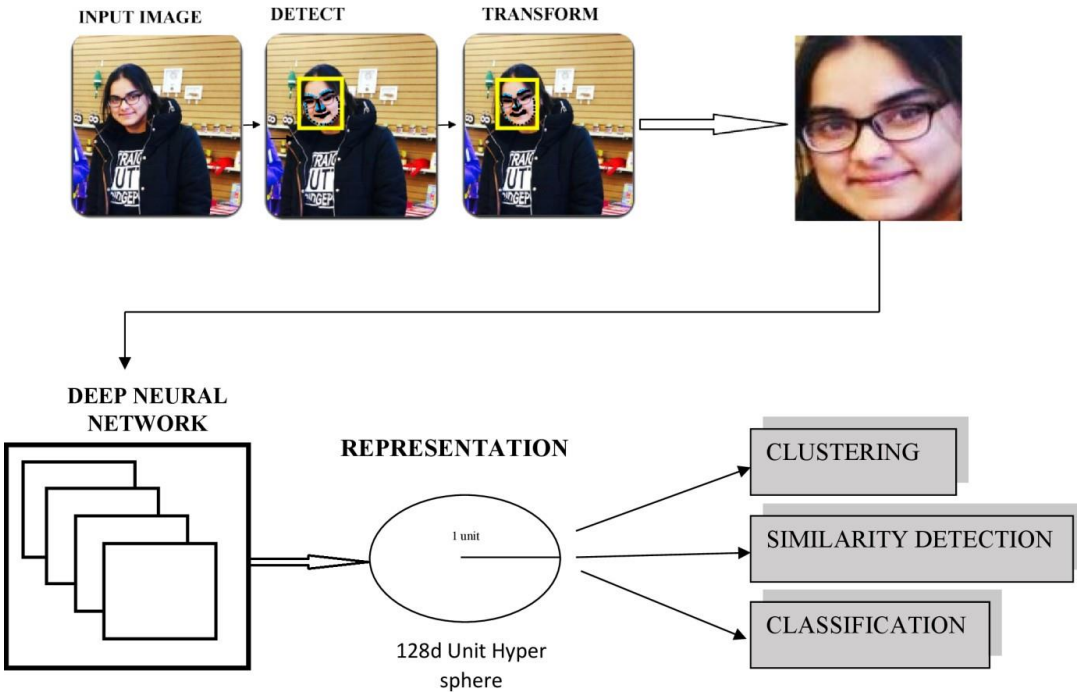


Figure 3.9: An overview of Face Recognition pipeline.

Quantifying faces in an image is the responsibility of the model Facenet which I have discussed above. First step in the Face Recognition pipeline is to input an image or the video stream to the pipeline on which face detection is applied to detect the location of the faces in the image or the video stream . Along with face detection, it also computes facial landmarks to preprocess image and align the image. The second step after the face detection and cropping of the image is to input the cropped image to our deep neural network which calculates the 128-d embeddings to quantify the face itself.

Figure 3.10: Deep Learning Face Recognition Model computing the Face Embeddings .

The input batch of the image includes three images which are the anchor image, the positive image and the negative image. Suppose the anchor image is our first image and has the identity I. And the second image is our positive image with the identity I as well i.e. it also contains the image of the person I. The negative and last image is the image that belongs to someone else who has the identity U. Now the batch is fed and neural network computes the 128-d embeddings for each face and then adjust the weights of the network with the help of triplet loss function such that the embeddings of the anchor image and positive image who

had the same identity lies closer to each other and the embeddings of the negative image with the identity U is pushed farther from these two. In this way, the network is able to learn to quantify faces and return discriminating embeddings for suitable for face recognition .

3.3 Data Design

The dataset I have contains three people i.e. myself, my friend and unknown (actress). Each class contains the total of six images (shown four) but I will suggest you to take at least 10-20 images per person you wish to recognize. Our face recognition dataset shown below.

Figure 3.11: Face Recognition dataset.

Chapter 4

IMPLEMENTATION

This chapter gives a brief description about implementation details of the system by describing each component with its code skeleton in terms of algorithm.

4.1 Deep learning with tensor flow (creating the environment).

To create the environment for our face recognition system, we need anaconda for python and CUDA toolkit. In anaconda, we either can create environment for the face recognition system to run on the CPU or the GPU. On my windows, I was able to fetch 14FPS and on my Mac book Pro, I was able to fetch 16FPS.

4.1.1 Facial Recognition in Images.

The first step is to extract the facial embeddings from the face dataset.

```
# load our serialized face detector from disk
print("[INFO] loading face detector...")
protoPath = os.path.join("face_detection_model/deploy.prototxt")
modelPath = os.path.join("face_detection_model/res10_300x300_ssd_iter_140000.caffemodel")
detector = cv2.dnn.readNetFromCaffe(protoPath, modelPath)

# load our serialized face embedding model from disk
print("[INFO] loading face recognizer...")
embedder = cv2.dnn.readNetFromTorch("openface_nn4.small2.v1.t7")
```

Taking the advantage of the “embedder” CNN, I extracted the embeddings.

```
# construct a blob for the face ROI, then pass the blob
# through our face embedding model to obtain the 128-d
# quantification of the face
faceBlob = cv2.dnn.blobFromImage(face, 1.0 / 255, (96, 96), (0, 0, 0), swapRB=True, crop=False)
```

```
embedder.setInput(faceBlob)
vec = embedder.forward()

# add the name of the person + corresponding face
# embedding to their respective lists
knownNames.append(name)
knownEmbeddings.append(vec.flatten())
total += 1
```

After extracting the facial embeddings, we will train our face recognition model. Now we have to train our standard model so that it would recognize the person based on the embeddings extracted by the embedder. After loading the facial embeddings and encoding our labels, I have trained the SVM model for face recognizing. After I have trained our models, the pickle files for model and encoder will be generated.

```
# train the model used to accept the 128-d embeddings of the face and
# then produce the actual face recognition
print("[INFO] training model...")
recognizer = SVC(C=1.0, kernel="linear", probability=True)
recognizer.fit(data["embeddings"], labels)
```

Now is the time to recognize faces with OpenCV, so after loading the three models, I will load the images and detect faces. In the output, I am drawing rectangle around the face and placing the text above the face which includes name and the probability. The more the images to train, the more the accuracy rate is.

The results are shown below.

Figure 4.1: The output shows that the face is with 53.83% probability

Figure 4.2: The output shows that the face is with 46.65% probability.

4.1.2 Face Recognition in live video stream using live webcam.

The face recognition basically follows the same procedure except for feeding in the image to recognize, we have used OpenCV to start capturing videos using the live webcam by initializing the "Video Stream" object. Also we have the system capturing the frames per second.

4.2 Image registration in Deep learning

4.2.1 Introduction

The process of aligning of the two images or more that belongs to the same scene is known as Image Registration. One of the images is called the base image and the other images are called the input images. The base image is the reference to which the other images are compared. The objective of the image registration process is to bring the input image into alignment after applying spatial transformation to it. In image processing, the introductory step is the process of the image registration. The major part of the research on medical image processing is covered by image registration . In medical field, the images that are created by different medical diagnostic modalities such as the process of Magnetic Resonance Imaging (MRI) or Single-Photo Emission Computed Tomography (SPECT) uses the image registration to compare various attributes in the images to see if the tumor is visible or not in the MRI or SPECT images . In computer vision, tasks like shape recovery, automatic change detection, automatic quality inspection, motion tracking, target template matching, are accomplished using image registration after alignment of the images has been achieved using different methods like Point based methods, Geometrical transformation, Surface or Intensity based method .

This part of the research is mainly focusing on Image Alignment using Geometrical transformation method. Face alignment helps in finding the geometrical structure of the faces in images and it also helps you in obtaining the canonical alignment of the face based on translation, scale and rotation . There are various types of face alignment process in which the model is pre defined and the transformation is applied to the image to compare the facial landmarks of the pre defined model and the image . The method for face alignment that I have implemented and followed in this research is based on facial landmarks which help in acquiring rotation, translation and scale representation of the face .

4.2.2 How it works

The process of face alignment is implemented using OpenCV and facial landmarks i.e. the input coordinates. The objective is to deform the input image and transform it to the output coordinate space after feeding in the input coordinates . The output coordinate space should consists of faces that are centered in the image, rotated in a way such that the eye lie in the horizontal line, and be scaled in a way that the sizes of the faces are almost identical. Thus, we can say that the process of face alignment is the type of data normalization. One can compare

face alignment with the process of normalizing a set of feature vector via zero centering or scaling to unit norm prior to training the machine learning model. I have achieved higher accuracy from my face recognition model after performing this process . The reason behind me performing this normalization is the other facial recognition algorithms like Eigenfaces, LBPs for face recognition, Fisher Faces and deep learning/metric methods. These methods aim at benefiting from applying facial alignment before even trying to identify the face .

4.2.3 Implementation

I started off by examining the FaceAligner implementation and learned about the insights that it provided. FaceAligner has the predictor; the desired output left eye position along with the width and height of the desired output face. It is composed of the align function which takes the image as an input and coverts the image into grayscale and produce a tight rectangle bounding box with the help of dlib face detector . After converting the landmarks into x, y coordinates, I will calculate the centre of the each eye and the angle between the eye centres.

This angle is the most important part which will help me align the face correctly after allowing me to rotate the image correctly. Now is the part where I actually aligned the faces. After setting up the environment and importing basic libraries, I have initialized the predictor and the aligner and the detector is initialized using dlib function.

```
scale = 4
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
fa = FaceAligner(predictor, desiredFaceWidth=256)
```

After this, we load the image and prepare it for face detection. Detecting faces in an input image is handled by dlib face detector which returns the list of bounding boxes around the faces. Then I have iterated through every bounding box and have displayed the original and aligned image. I have also stored the aligned image in the output folder.

```
for rect in rects:

(x, y, w, h) = rect_to_bb(rect)
faceOrigin = imutils.resize(image[y:y + h, x:x + w], width=256)
faceAligned = fa.align(image, gray, rect)
cv2.imwrite(os.path.join(path , '01.jpg'), faceAligned)
cv2.imshow("Original", faceOrigin)
cv2.imshow("Aligned", faceAligned)
cv2.waitKey(0)
```

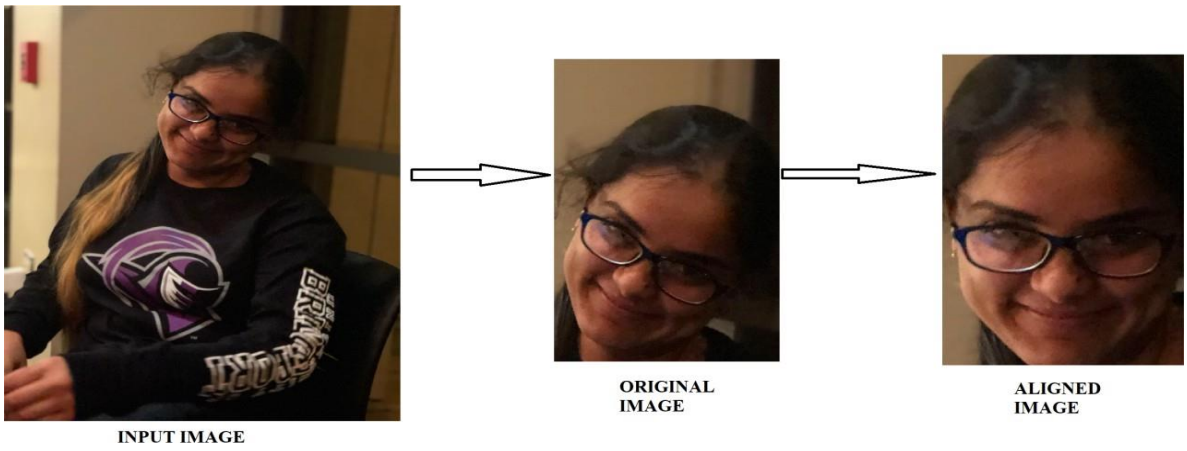


Figure 4.3: Shows the alignment of the image.

4.3 Yolo object detection using deep learning and openCV

4.3.1 Introduction

You Only Look Once (YOLO) Object Detection in deep learning is trained on a loss function that directly corresponds to detection performance and the whole model is trained jointly . When we look at something, we suddenly come to conclusion about the things presented to us. Suppose if we look at an image, our brain instantly recognizes things and objects present in the image along with their behavior. Like human brain, which is fast and accurate, there are various algorithms for object detection that allows computers to drive cars without human intervention . To detect an object in an image or real life, the classifiers that are used by current detection system, takes the classifier for that particular object and evaluate it at various scales and location in the testing image . Models like DPM i.e. deformable parts model use the approach of sliding window in which the classifier is run at evenly spaced locations over the whole image .

R-CNN use the approach that is based on region proposal methods. This method first generates potential bounding boxes in an image and then run the classifier on these proposed boxes. After the classification, post processing is done on the image to refine the bounding boxes which will help in removing the duplicate bounding boxes and detection and rescore the boxes based on other objects present in the image .But these complicated pipelines are really slow and they are also hard to optimize as each component is trained separately . On the contrary, YOLO uses reframing of the object detection as a single regression problem in which object detection is accomplished using image pixels to bounding boxes and class probabilities for those boxes .

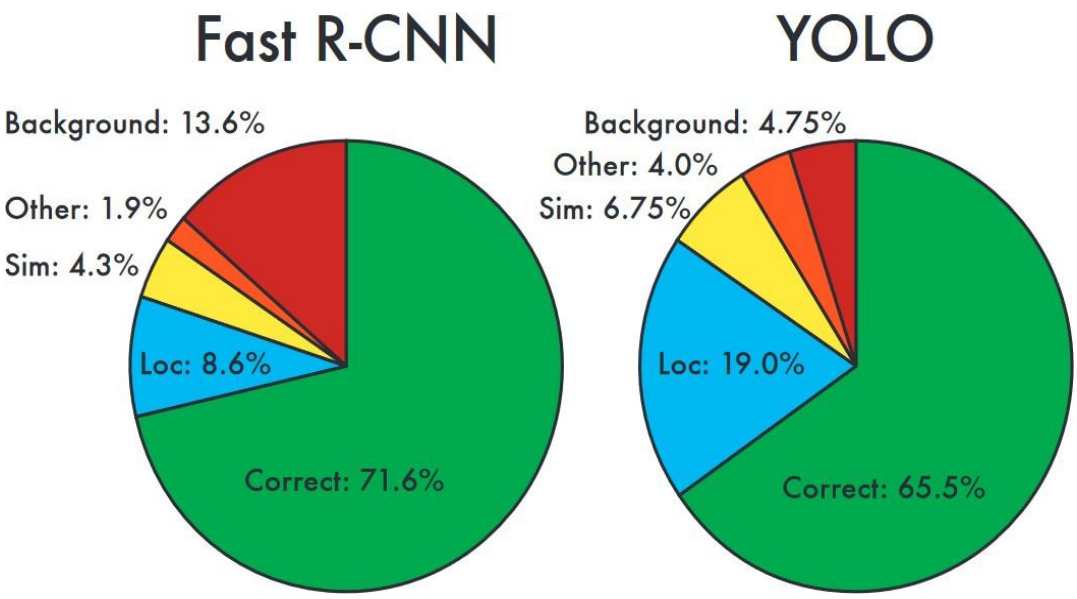


Figure 4.4: Shows the error analysis for Fast R-CNN vs. YOLO

4.3.2 How it works

YOLO is easy and extremely fast. Unlike sliding window technique region proposal based technique, YOLO inspects the entire image during training and the testing time so that it would be able to encode contextual information about the classes as well as their appearances . Fast R-CNN, a top detection model , mistakes background patched in an image as an object as it is unable to see the broader context in an image because it doesn't follows the technique used by YOLO to scan the entire image at once. YOLO makes less than half the number of errors in background patches compared to fast R-CNN.

The YOLO design does the end to end training and activates real time speed while maintaining high average precision . The system divides the input image into an s x s grid and if the centre of the object present in the image falls into the grid cell, that grid cell becomes responsible for detecting that object. Each grid cell is responsible for detecting bounding boxes and the confidence for those boxes . Suppose if there is no object present in the cell then the confidence score for that cell should be zero. Each bounding box includes the five predictions and the confidence . At the time of the test, the YOLO model, multiplies the conditional class probabilities and the individual box confidence predictions which further gives us class specific confidence score for each box . Then the encoding is done for both the probability of the class appearing the box and how well the predicted box fits the object .

$$\Pr(\text{Class}_i|\text{Object}) * \Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) * \text{IOU}_{\text{pred}}^{\text{truth}}$$

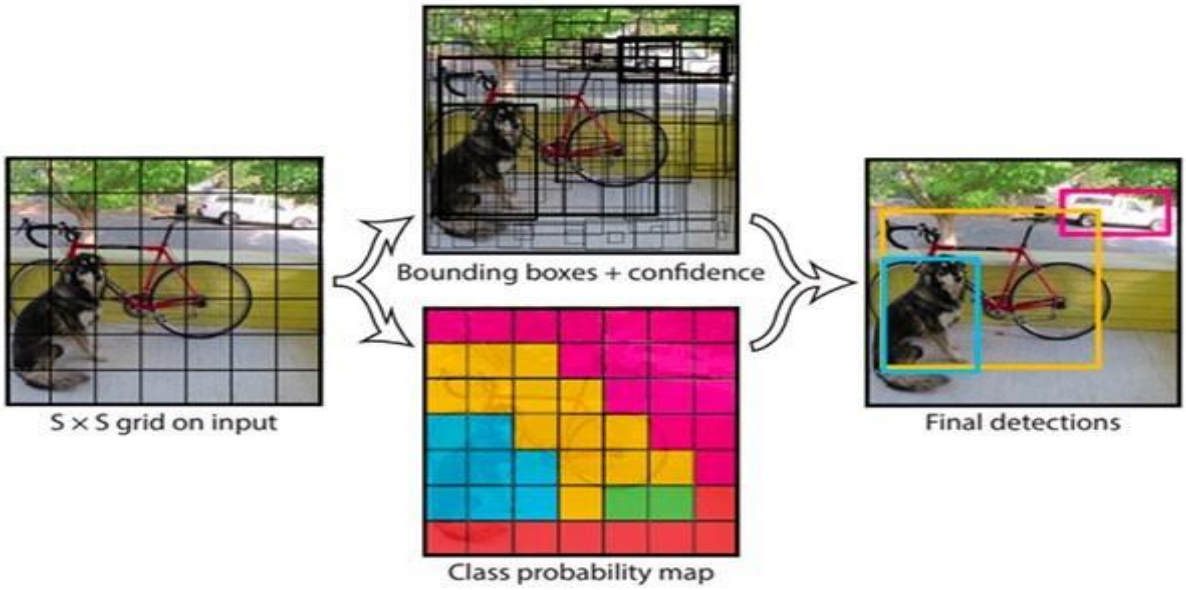


Figure 4.5: A simplified illustration of YOLO object detector pipeline.

4.3.3 Implementation

I have started by loading the coco class labels and set random colors for each class.

```
# load the COCO class labels our YOLO model was trained on
labelsPath = os.path.join("coco.names")
LABELS = open(labelsPath).read()
```

Now I will load the weights and configuration models followed by loading the image and sending it to the network which determines the output layer names from the YOLO model and further construct the blob from the image . After the blob is prepared, the forward pass through the YOLO network is done and the interference time for the YOLO is displayed on the console.

```
weightsPath = os.path.sep.join("yolov3.weights")
configPath = os.path.sep.join("yolov3.cfg")

# load our YOLO object detector trained on COCO dataset (80 classes)
print("[INFO] loading YOLO from disk...")
net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)
```

Now I have initialized the lists of detecting bounding boxes, confidence and the class IDs after which I have populated all of them with the YOLO data from output layers. After

looping over each output layer and detected boxes in output, I have extracted the class IDs and the confidence to apply it to filter out weak detections.

```
for output in layerOutputs:
    # loop over each of the detections
    for detection in output:
        # extract the class ID and confidence (i.e., probability) of
        # the current object detection
        scores = detection[5:]
        classID = np.argmax(scores)
        confidence = scores[classID]

        # filter out weak predictions by ensuring the detected
        # probability is greater than the minimum probability
        if confidence > 0.5:
        # scale the bounding box coordinates back relative to the
        # size of the image, keeping in mind that YOLO actually
        # returns the center (x, y)-coordinates of the bounding
        # box followed by the boxes' width and height
        box = detection[0:4] * np.array([W, H, W, H])
        (centerX, centerY, width, height) = box.astype("int")

        # use the center (x, y)-coordinates to derive the top and
        # and left corner of the bounding box
        x = int(centerX - (width / 2))
        y = int(centerY - (height / 2))

        # update our list of bounding box coordinates, confidences,
        # and class IDs
        boxes.append([x, y, int(width), int(height)])
        confidences.append(float(confidence))
        classIDs.append(classID)
```

After filtering out unwanted detections, I have scaled bounding box coordinates so that I can display them properly on my original image. Then I have pulled out the coordinates and dimensions of the bounding boxes and have used all this information to detect the top left corner of the bounding boxes to further update the boxes, confidences and the class IDs. Now I have applied the non maxima suppression which YOLO doesn't apply itself, which

keeps only the most confident bounding boxes. Finally I have looped over IDXS which was determined by non maxima suppression.

Figure 4.6: Shows the results of the YOLO detection

4.4 Image captioning

Scene understanding is one of the fundamental, but also most difficult tasks of computer vision and ability to automatically generate text captions of an image or video can have a great impact on lives of many. However, it is much more complicated than simple clas- sification or object recognition tasks, because the model also need to understand relations between the recognized objects and encode that relationship correctly in the caption.

Two main approaches to image captioning were popular, until neural networks dominated the field. The first one used caption templates, which were filled by detected objects and relations. Second was based on retrieval of similar captions from database and modifying them to fit the current image. Question of similarity ranking has been addressed by many papers, which are based on the idea of joint embedding vector space for both images and captions , as it transforms estimation of similarity to a simple proximity measurement. Both approaches included a generalization step to remove information relevant only to current image, for example names.

These approaches were quite successful in describing images, but they are heavily hand-designed. Also their text-generation power is fixated on the database/embeddings and is not able to describe previously unseen compositions of objects. Over time these approaches fell out of favor, as methods leveraging the power of neural networks emerged.

However, some of their ideas proved to be useful in the new environment and we can encounter them in recent works . Many of the new methods, which use neural nets, are inspired by the success in training recurrent nets for machine translation. It is worth mentioning Sutskevers work , which studied general sequence to sequence mapping by converting an input sequence to the fixed length vector, which is then decoded to the output sequence. This encoder–decoder architecture is closely related to the autoencoders and work of Kalchbrenner and Blunsom , who were first to map the entire input sequence to vector. The introduced encoder–decoder architecture is important to the captioning task, because image description problem can be interpreted as a translation from an image to a sentence. In this case, encoder part of the model is usually a convolutional neural net, as they are excellent in the image classification .

Decoder part is the similar to the one in machine translation models – an RNN or a type of LSTM, as the output for both tasks is essentially the same. Following the encoder–decoder idea, current image captioning research is shifting towards models, which are trained end-to-end with some type of stochastic gradient descent (SGD) algorithm. The reasons for the shift can be simplicity and a lot less hand design than in other methods. Different type of the state-of-the-art models are based on proven pipeline of key-word detection, sentence generation, and ranking, which exploit the power of embedded neural networks, which specialize in single task.

This approach is more closely described in section discussing article From Captions to Visual Concepts and Back. The current field is consolidating, thanks to MS COCO Captioning Challenge1 and dataset created for it. Simple public access to the necessary data makes model creation easier and the best researchers can compete directly against each other by using MS COCO evaluation server. In the rest of this section, I describe several works, which were submitted to the challenge in 2015 and had the best performance.

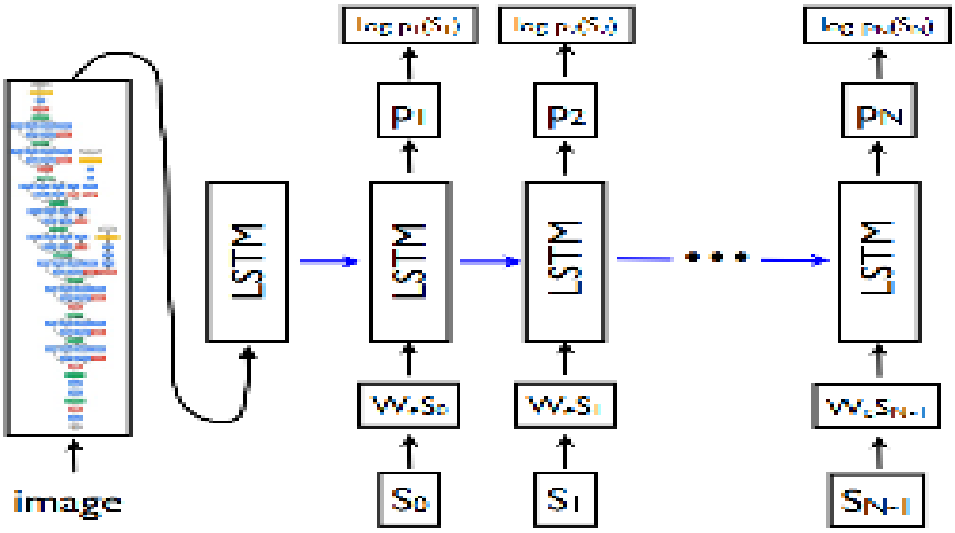
MS COCO dataset is described, together with other datasets Show and Tell: A Neural Image Caption Generator Show and Tell is model created by Google researchers, which tied for the first place in MS COCO Captioning Challenge with the following model From Captions to Visual Concepts. The main idea of this work is to use recent advancements in machine translation and apply them for image captioning. Model uses encoder–decoder architecture, with CNN for the encoder part and RNN for the decoder part, as described earlier. Model is

trained to maximize the likelihood p(S|I) of producing a target sequence of words S = S1, S2, ... given an input image I Used convolutional neural net has been pre-trained for an image classification task and last hidden layer of this network has been used as an input to the RNN decoder. The RNN part of the network is made of LSTM units based on following equations:

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1}) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1}) \\
\widetilde{C}_t &= \tanh(W_{xc}x_t + W_{hc}h_{t-1}) \\
C_t &= f_t \odot C_{t-1} + i_t \odot \widetilde{C}_t \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1}) \\
h_t &= o_t \odot C_t
\end{aligned}
$$

The language model is working on the word-level, part of the RNN is word embed- ding , which is trained together with the model. CNN, which is used to generate a configuration vector from the image, is connected to the RNN at the beginning as the first input before the generated sequence. Overall structure of the model is visualized on Figure 3.1 and can be represented by following equations:

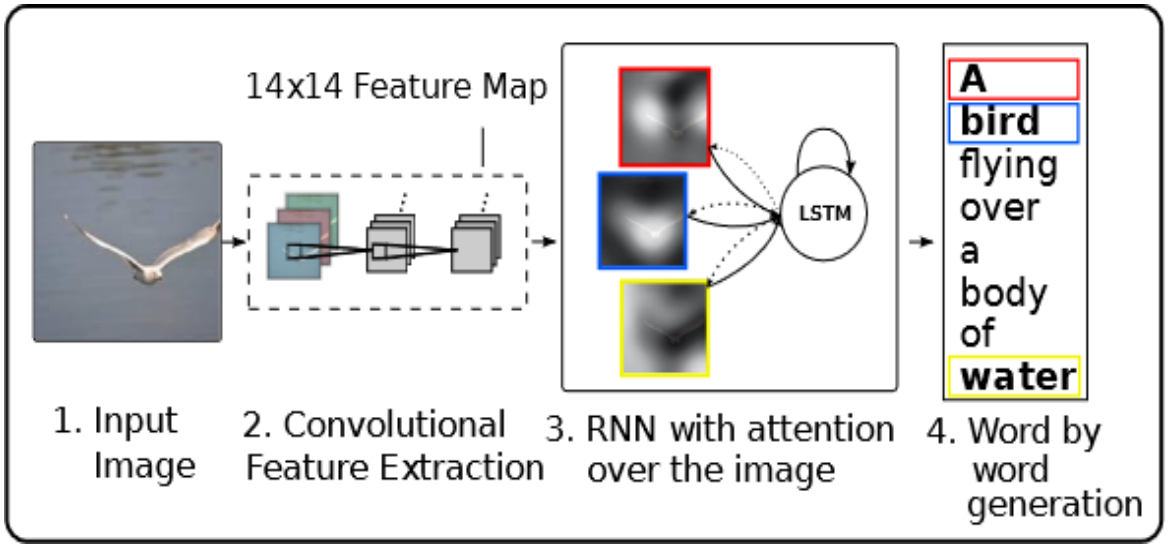
$$
\begin{aligned}
x_{-1} &= CNN(I) \\
x_t &= W_e S_t \qquad t \in \{0 \ldots N-1\} \\
p_{t+1} &= LSTM(x_t) \qquad t \in \{0 \ldots N-1\}
\end{aligned}
$$



As the image and word encodings are used in the same way, model is effectively mapping both images and words into the same vector space. During the sequence input, special start word and stop word signated to mark start and end of the sequence are used.

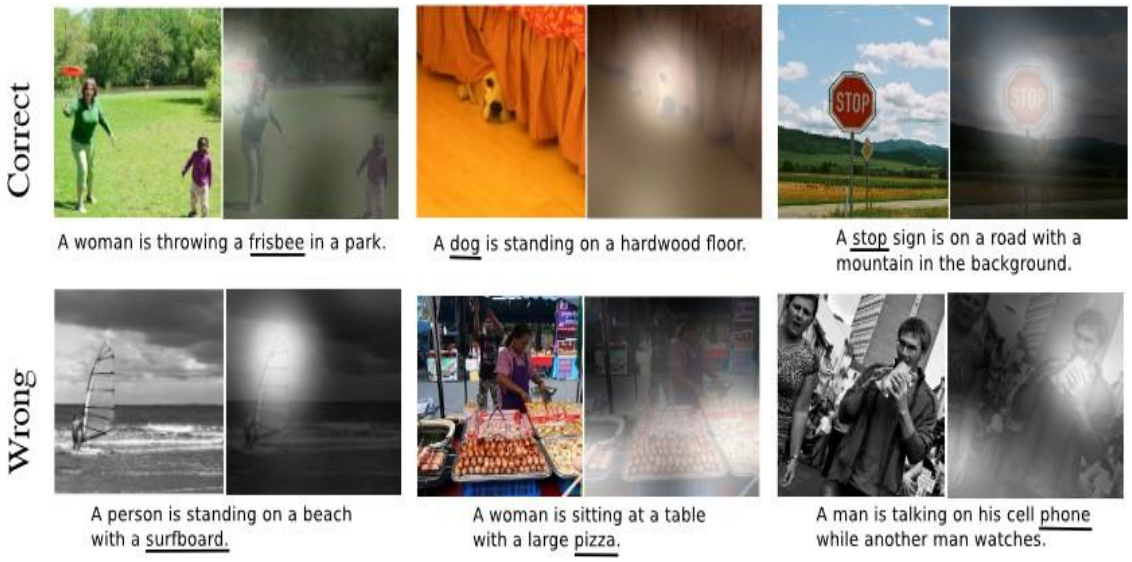
The CNN component of the model has been initialized to an ImageNet trained model, which helped quite a lot in terms of generalization. Word embeddings were left uninitialized (initialized randomly) as they did not observed significant gains while using large corpus. Dropout and ensebling used during training gave minor improvements. Model has been trained using SGD with fixed learning rate and no momentum. For the embeddings vector and the LSTM memory 512 dimensions were used. During the inference, beam search has been used to improve the results. From Captions to Visual Concepts and Back This work took quite a different approach than a previous one, however, both tied for the first place in the captioning competition. This model is not trained end-to-end with a single training algorithm rather it has three connected stages. Full pipeline of the model is on the Figure and its description follows. First, model learns to extract nouns, verbs, and adjectives by applying CNN to regions of the image. These words come from the vocabulary constructed with 1000 most common words in the training captions. By running detector on the image regions, model is able to produce a bag of bounding boxes, which represent the location of the appropriate word in the image. Network used for the word detection is the 16-layered CNN, commonly referred as VGGnet, from a conference paper .

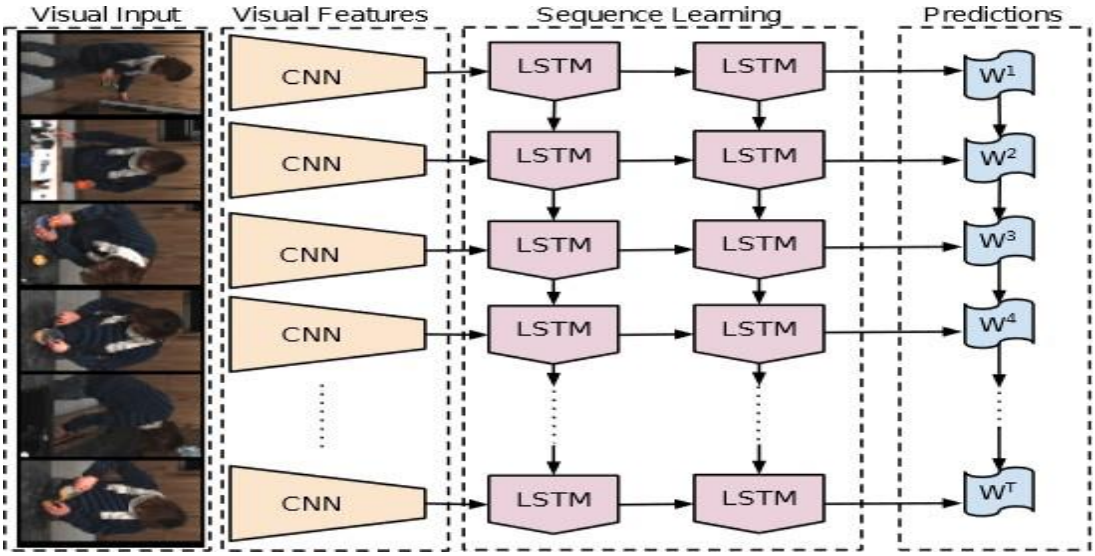
Second stage use the extracted words to guide a language model to generate sentences likely to describe the image. The maximum entropy language model estimates the probability of a word conditioned on the preceding words as well the words with high likelihood detections, yet to be mentioned. This encourages all the words to be used, while avoiding repetitions. A left-to-right beam search is used during generation. After extending each sentence with a set of likely words, the top sentences are retained and the others pruned away. The process continues until a maximum sentence length is reached. In the third stage, candidate captions are re-ranked using Minimum Error Rate Training (MERT) and the best one is selected. MERT uses a linear combination of features computed over the sentence, for example log-likelihood of the sequence or its length. One of the features is Deep Multimodal Similarity Model (DMSM) score, which measures similarity between images and text. The DMSM has been proposed in this paper and the model trains two neural networks that map images and text fragments to a common vector representation. These vectors are used to compute the cosine similarity score, which is then sent to MERT. As mentioned earlier, this model is the state-of-the-art on the MS COCO Captioning Challenge, as it tied for the first place with the Show and Tell work. Their final and best performing model used VGGnet, word detector score in maximum entropy language model, proposed DMSM and use fine-tuned VGGnet features. According to human judgment, generated captions are equal to or better than human-written captions 34% of the time. Direct comparison of the From Captions to Visual Concepts approach with the Show and Tell, which was presented earlier, is in the additional article by the same authors. They examine the issues of both approaches and achieve state-of-the-art performance by combining key aspects of RNN and maximum entropy methods.



Show, Attend and Tell: Neural Image Caption Generation with Visual Attention Show, Attend and Tell is method, made by researchers from universities in Toronto and Montreal, which introduced an attention based model. Attention is one of the most interesting parts of the human visual system. Rather than compressing an entire image into a static representation, attention allows for salient features to dynamically come to the forefront as needed. Proposed model has encoder-decoder architecture with feedback connections for attention. Overall structure of the model is on Figure 3.3. Encoder part use a CNN to extract set of feature/annotation vectors (not just one). Each of the vectors correspond to a part of image. To obtain a correspondence between them, features from a lower convolutional layer are used. Decoder part is a LSTM network working of the word level, which generates, apart from the word of the output, a context vector - a dynamic representation of the relevant part of the image at time. The paper explored two attention mechanisms computing the context vector from the annotation vectors. First is the stochastic “hard” mechanism, which interprets the values in the context vector as the probability that corresponding location is the right place to focus, while producing the next word. Second is deterministic “soft” mechanism introduced in , which gives the relative importance of the location by blending values of all annotation vectors together. This method is fully trainable by the standard SGD methods. Article also shows how we can gain insight and interpret the results of the model by visualizing where and on what was the attention focused. Examples of the attention visualizations with both, correct and wrong generations, are on Figure 3.4. Visualizations show that the model can attend even a “non-object” regions. This adds an extra layer of interpretability to the output. The model learns alignments that correspond very strongly with human intuition and, even, in the cases of mistakes, we can understand why the captions were generated. Similarly to previous article, Show, Attend and Tell used VGGnet , a CNN trained on the ImageNet,

which was not fine-tuned. Model was trained with several algorithms and researchers found that for Flickr8k dataset, RMSProp worked best, while for Flickr30k and MSCOCO datasets, Adam algorithm was used. Performance during training was also improved by creating mini batches of sentences with same length, which greatly improved convergence speed.





Long-term Recurrent Convolutional Networks for Visual Recognition and De- scription The research group from Berkeley presented Long-term Recurrent Convolutional Networks (LRCN) , which combines network with convolutional and long-range temporal layers for several tasks. It is possible to apply LRCN to recognize activity performed on the video (sequential input → fixed output), generate description of the image (fixed input → sequential output) or

describe video (sequential input → sequential output). Architecture of the proposed model, is similar to the Show, Attend and Tell, as image is processed by CNN and sent to the input of the LSTM in each time step. Feedback attention connections are missing in this model. According to the task specification, method can use separate convolutional networks, different for each time step with specific input, or single CNN throughout all the time steps. LRCN tied with Show, Attend and Tell in MS COCO Captioning Challenge for the third place. However, this does not mean the models are equal in general performance, as each of them focuses on different research topic. Datasets Large amounts of data are necessary requirement in training deep neural nets like CNNs and RNNs, as well as sufficient computing power. Access to the machines and hardware suitable for training has been made in recent years extremely easy, with the rise of virtualization services. Obtaining enough data is different issue and it is currently the biggest problem. Especially, creation of image captioning datasets is quite complicated. As there is no automatized way to generate data, all the image descriptions have to be human-generated. This is one of the reasons, only few specialized datasets are created. There are two main options how to get images and captions. First way is, by using user- generated data from an online service, most commonly Flickr2. However, these captions are not made specifically for the task and could be prone to error. Second option is to gather only images, again from Flickr or other online services, and create captions for direct use in the dataset manually. Amazon Mechanical Turk3 is heavily used for this task. Following Table 3.1 lists the most popular datasets. All these datasets were created directly for the image captioning task, with captions generated through Amazon Mechanical Turk. Flickr8k , from 2013, was one of the first datasets created for this purpose. It has been later expanded into Flickr30k . The biggest dataset is Microsoft Common Objects in Context (MS COCO) , created for the MS COCO captioning challenge. CIDEr datasets PASCAL-50S, ABSTRACT-50S are youngest mentioned, designed specifically for evaluation with the CIDEr metric discussed

In this section, we want to foster a language model which can make the subtitle reliable with the picture content. The current encoder-decoder model and its variations, which are the most well known models for picture inscribing, utilize the picture highlights in three ways:

- they infuse the encoded picture highlights into the decoder just a single time at the underlying advance, which doesn't empower the rich picture content to be investigated adequately while bit by bit producing a text subtitle;
- they connect the encoded picture highlights with text as additional contributions at each progression, which presents superfluous commotion; and
- they utilizing a consideration instrument, which expands the computational intricacy because of the acquaintance of extra brain nets with recognize the consideration districts.

Name	Images	Captions per image	Note
MS COCO[4]	123 287	5	Images are divided - 82 783 for training and 40 504 for testing purposes.
Flickr30k[5]	31 783	5-6	An extension of Flickr8k dataset.
Flickr8k[6]	8 092	5	Focused on people or animals (mainly dogs) performing some specific action.
PASCAL-50S[7]	1 000	50	Built upon images from the UIUC Pascal Sentence Dataset.

Figure 4.7: Image captioning datasets

Not the same as the current techniques, in this section, we propose an original organization, Recall Network, for producing inscriptions that are predictable with the pictures. The Recall Network specifically includes the visual highlights by utilizing a GridLSTM and in this manner can review picture substance while producing each word. By bringing in the visual data as the inert memory along the profundity aspect LSTM, the decoder can concede the visual highlights.

progressively through the inborn LSTM structure without adding any extra brain nets or boundaries. The Recall Network effectively keeps the decoder from straying from the first picture content. To confirm the productivity of our model, we directed thorough examinations on full and thick picture inscribing. The analysis results obviously show that our Recall Network outflanks the ordinary encoder decoder model by a huge degree and that it performs equivalently to the cutting edge strategies.

4.4.1 Presentation

Late advances have uncovered that encoder-decoder systems [32, 81, 95] can accomplish start to finish preparing and are fit for conveying the picture subtleties in a single sentence. These brain network-based strategies are propelled by ongoing advances in machine interpretation. In these structures, an encoder network, a Convolutional Neural Network (CNN), encodes a picture into a setting vector. Then, the decoder organization, a Recurrent Neural Network (RNN), translates the setting vector into a grouping of words. For the most part, parts, for example, LSTM or GRU can be utilized in the RNN language models. Albeit the traditional

encoder-decoder model gives a functional method for building the visual and text information stream, it actually has a few constraints on picture predictable subtitles. The picture highlights in this model are infused just a single time into the decoder at the underlying advance, and that implies that the decoder might go astray from the picture substance while step by step producing the text inscription. Subsequently, the rich visual subtleties can't be adequately considered. Albeit, in , the picture highlights are linked with word inserting and are taken care of into the decoder as additional contributions at each time step, the presentation isn't fulfilling, for the most part because of the superfluous clamor included . Most as of late, the consideration encoder-decoder models [95] [99] have been acquainted with tackle the bottleneck in . In any case, the consideration system needs to depend on extra brain nets to underline the consideration locales. In a perfect world, the visual data ought to be thought about constantly and appropriately for producing each word.

In this part, to manage the lacks in the current encoder-decoder systems, we propose a clever organization, Recall Network, for producing picture predictable subtitles. The principle objective of the Recall Network is to review the visual data consistently and appropriately while producing each word. We are enlivened by GridLSTM which has LSTM cells in various aspects. In , it has been observed that having LSTM units along the profundity aspect is more powerful than not having them in some normal language handling assignments. To accomplish our objective, we build the organization with the GridLSTM and change the GridLSTM to adaptively consider picture highlights through memory cells of the profundity LSTMs. In this organization, the picture highlights are at first contribution as an outline of the entire picture content and afterward reviewed ceaselessly at each progression as the "past memory" along the profundity aspect in the GridLSTM. Rather than generally conceding the still visual data, we utilize the memory cell in the profundity LSTM design to specifically neglect and update the visual data as per the relating word. The plan yields a basic yet productive organization that adaptively includes the visual data at each progression without expanding any additional nets or teachable boundaries.

The fundamental commitments of this part can be summed up as follows:

- To keep the decoder from digressing from the visual direction, we present an inventive encoder-decoder model that reviews the picture data persistently in the decoder to produce picture reliable subtitles.
- We alter the GridLSTM to adaptively consider picture highlights through the profundity LSTM memory cell, which accomplishes powerful usefulness without additional nets or boundaries being presented.

- Comprehensive examinations are directed on both full picture subtitling and thick inscribing. Our Recall Network beats the traditional encoder-decoder model by a huge degree and performs equivalently to the-condition of-workmanship techniques.

4.4.1.1 LSTM Background

In this section, we briefly introduce the background of LSTM. A Recurrent Neural Network is a neural network that processes a sequence of entities, while a Long Short-Term Memory (LSTM) performs as an activation function unit in the RNN. The conventional RNN processes the input sequence X = (x1,· · ·,xT) and output sequence Y = (y1,· · ·,yT) through an internal hidden state h as follows:
10pt]

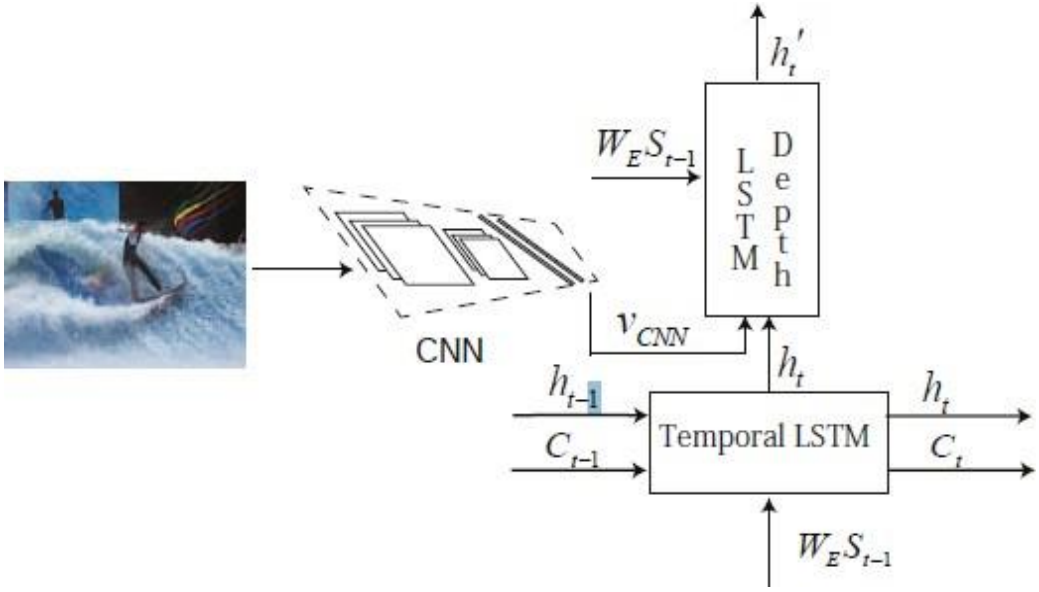
ht = g (ht1,xt),

log p(^yt|x<t) = f (ht),

where g is a nonlinear activation function parameterized by a set of parameters . p(^yt|x<t) is the probability of prediction ^yt at time t. f can be a parametric function learned jointly in the whole framework. The conventional RNN encounters a problem of vanishing gradient, where the backward signals may decay sharply through the chain. A more sophisticated activation function, i.e. the LSTM, is introduced to avoid this problem through gating and memory cells. The LSTM updates and outputs as follows:

$$\mathbf{i}_t = \sigma(\mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{W}_i\mathbf{x}_t + \mathbf{b}_i),$$
$$\mathbf{f}_t = \sigma\left(\mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{W}_f\mathbf{x}_t + \mathbf{b}_f\right),$$
$$\mathbf{o}_t = \sigma(\mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{W}_o\mathbf{x}_t + \mathbf{b}_o),$$
$$\tilde{\mathbf{C}}_t = \tanh(\mathbf{U}_C\mathbf{h}_{t-1} + \mathbf{W}_C\mathbf{x}_t + \mathbf{b}_C),$$
$$\mathbf{C}_t = \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{C}}_t,$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{C}_t),$$

where it, ft, and ot are the input gate, forget gate, and output gate, respectively. Ct is the memory cell. W, U and b are parameter metrics to be learned. The LSTM mechanism has two important properties. The memory cell is obtained by a linear transformation of the previous memory cell and gates, rather than a nonlinear function, which ensures that the signal in backward propagation does not decay sharply. The memory cell also involves new information to update, and drop information that is selected to be forgotten, which acts as an attention system Structure Overview An outline of our methodology is portrayed in Fig.. Our model depends on the regular encoder-decoder model. Instead of just utilizing worldly LSTM units to display the likelihood of the objective words, we further endeavor the profundity LSTM to associate with picture sees. In machine interpretation, a sentence in a source language can be encoded into a setting vector. Then, the decoder network makes an interpretation of it into a sentence in the objective language. In the ordinary encoder-decoder system for picture subtitling, given a picture I, the Figure: delineation of the proposed model. Our model contains two LSTMs at each progression. We utilize the worldly LSTM to communicate the consecutive inscription states and the profundity LSTM to incorporate the visual data. model will "make an interpretation of" it into a subtitle that can be addressed as a succession of words similarly:



where S0 is a START token added in pre-handling. The goal is to boost the amount of the log probability of the relating words:

$$\mathbf{S} = [\mathbf{S}_0, \mathbf{S}_1, \mathbf{S}_2, \cdots \mathbf{S}_N],$$

where addresses the boundaries to be learned. The picture I is encoded as a setting vector, vCNN, with a Convolutional Neural Network. Then the setting vector is infused to a Recurrent Neural Network and the RNN decoder produces a subtitle word by word. Typically, RNN comprises of an arrangement of LSTM units. At each progression, with the LSTM ouput, we can get the probabilities over the words in the word reference. Each word is addressed by a word inserting vector , WESt, where WE means a weight metric for the word implanting. The word St is a pointer one-hot vector as 1×V, where V is the size of the word reference. It comprises of 0 in every one of the positions, and it is set to 1 if the ith position is utilized to distinguish a word. In the customary encoder-decoder system, the log probability (the restrictive likelihood is improved as pt in Fig. 4.2) is displayed as:

$$\theta^* = \arg\max_{\theta} \sum_{t=1}^{N} \log p(\mathbf{S}_t | \mathbf{I}, \mathbf{S}_0, \cdots, \mathbf{S}_{t-1}; \theta),$$

$$\log p(\mathbf{S}_t | \mathbf{I}, \mathbf{S}_0, \cdots, \mathbf{S}_{t-1}) = f(\mathbf{h}_t),$$

where f is a nonlinear capacity that yields the likelihood of St. SoftMax work is taken advantage of here. ht is the secret state as well as the result in the Recurrent Neural Network. With the LSTM embraced, ht is demonstrated as:

$$\mathbf{h}_t = LSTM(\mathbf{x}_t, \mathbf{h}_{t-1}, \mathbf{C}_{t-1}),$$

where xt is the info. Review the Visual Information Continually in the Decoder In the traditional encoder-decoder structure, the decoder interprets the picture highlights, vCNN, at the initial time step to give the underlying state to the ensuing time steps. The regular encoder-decoder system creates each word just in view of the past state and the past ground truth word or the recently produced word assuming that planned testing is taken advantage of. The contributions to the LSTM can be addressed as:

$$\mathbf{x}_t = \begin{cases} \mathbf{v}_{CNN} & t = 0 \\ \mathbf{W}_E \mathbf{S}_{t-1} & t > 0 \end{cases}.$$

The GridLSTM orchestrates the LSTM calculation in a multi-faceted matrix. We make utilization of this property to keep the consecutive inscription structure and present the visual direction all the while. Subtleties on the GridLSTM-based calculation are introduced, including how to process it and why we use it.

$$\log p(\mathbf{S}_t | \mathbf{I}, \mathbf{S}_0, \cdots, \mathbf{S}_{t-1}) = f\left(\mathbf{h}'_t\right),$$

The GridLSTM orchestrates the LSTM calculation in a multi-faceted matrix. We make utilization of this property to keep the consecutive inscription structure and present the visual direction all the while. Subtleties on the GridLSTM-based calculation are introduced, including how to process it and why we use it.

4.4.1.2 Review with the Depth Dimension LSTM

Naturally, a produced word ought to depend on visual data and language imperatives adaptively. For instance, the picture data has little impact when the RNN decoder produces non-visual words, for example, "of" and "the". A few produced words clearly depend on the language model, for example, "telephone" in the wake of "chatting on cell". In a few different circumstances, the picture data is significant for the age, for example, perceiving an article as a "canine" or a "man". In late advances, the twofold LSTM configuration has been exhibited to be successful for arrangement age. Roused by , as opposed to connecting the picture

elements and word installing highlights, we utilize a fleeting aspect LSTM (tLSTM) to send the successive subtitle states and utilize a profundity layered LSTM (dLSTM) to coordinate the visual data while creating the inscriptions. At the initial phase in the age grouping, the visual vector is treated as the tLSTM's feedback, and the tLSTM gets word installing as the info subsequently. At each progression, the visual vector is reviewed/treated as the past memory cell in the dLSTM. tLSTM and dLSTM share boundaries. The calculation subtleties are introduced in. The principal LSTM (the fleeting LSTM), works precisely equivalent to the regular LSTM in Equation 7.11. Acquiring the states in the transient aspect, ⟨Ct,ht⟩, we further utilize ht in the profundity aspect. On the profundity aspect, ht fills in as the underlying secret state to give the setting data, while WESt1 functions as the info again to give the text direction. The info door chooses how much another subject will be added in, while the neglect entryway influences the degree to which a past subject ought to be neglected.

$$\mathbf{i}_t' = \sigma(\mathbf{U}_i\mathbf{h}_t + \mathbf{W}_i\mathbf{W}_E\mathbf{S}_{t-1} + \mathbf{b}_i),$$

$$\mathbf{f}'_t = \sigma\left(\mathbf{U}_f\mathbf{h}_t + \mathbf{W}_f\mathbf{W}_E\mathbf{S}_{t-1} + \mathbf{b}_f\right).$$

The visual data, vCNN, is reviewed as the past memory cell in the profundity aspect. Through refreshing the memory cell:

$$\mathbf{C}_t' = \mathbf{f}_t' \odot \mathbf{v}_{CNN} + \mathbf{i}_t' \odot \tanh(\mathbf{U}_C\mathbf{h}_t + \mathbf{W}_C\mathbf{W}_E\mathbf{S}_{t-1} + \mathbf{b}_C),$$

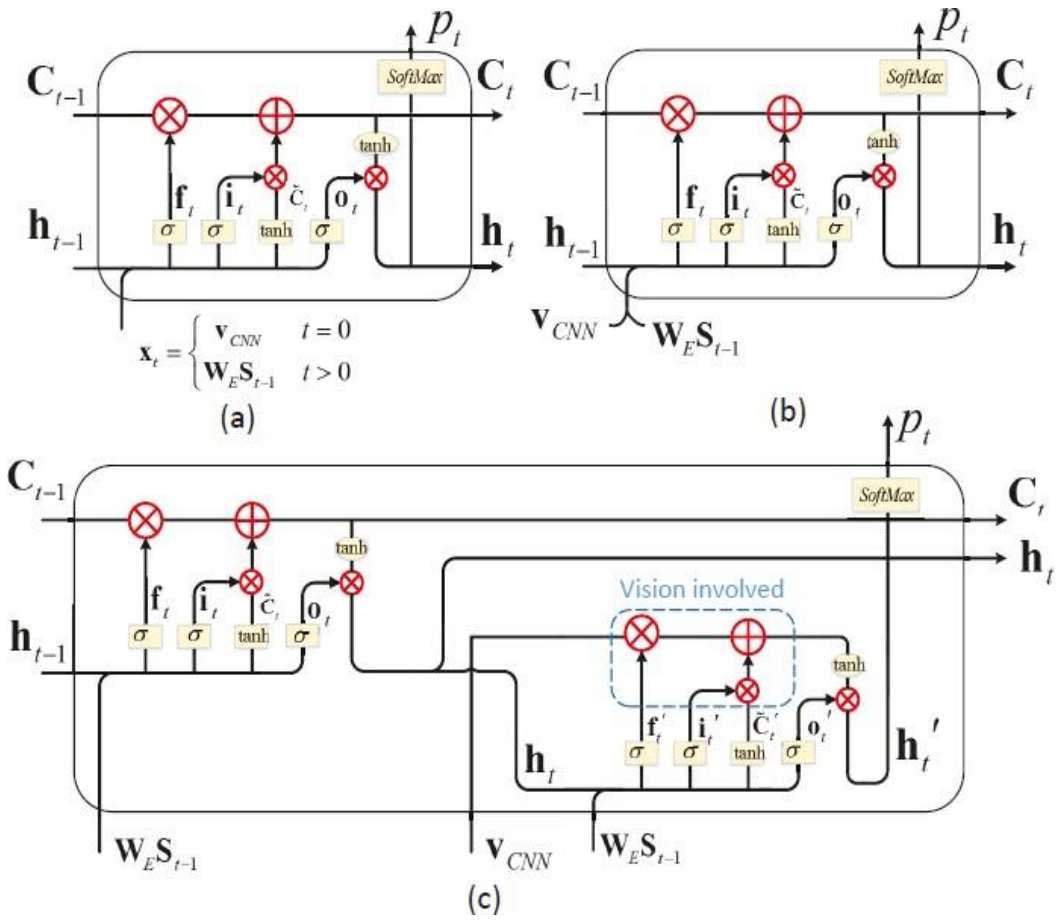


Figure : Three different decoder units. (a): The image representation is only processed once with the LSTM. (b): The image representation and word embedding are concatenated as extra inputs for the LSTM. (c): Our Recall Network. some visual information is selectively forgotten through the forget gate that is formulated by the input text, WESt1, and the temporal hidden state, ht. The adaptive visual information is then integrated with the new subject into Ct . Finally, with the output gate, we obtain ht.

$$\mathbf{o}_t' = \sigma(\mathbf{U}_o\mathbf{h}_t + \mathbf{W}_o\mathbf{W}_E\mathbf{S}_{t-1} + \mathbf{b}_o),$$

$$\mathbf{h}_t' = \mathbf{o}_t' \odot \tanh(\mathbf{C}_t').$$

As per the LSTM inside component , the info door controls the degree to which another worth streams into a cell, and the neglect entryway controls the degree to which a worth remaining parts in a cell. The memory cell interfaces with the information entryway as well as the neglect door, where it consolidates new data to refresh and drops specific past data to

neglect. In our model, the visual data is reviewed as the past memory cell in the profundity aspect. Through refreshing the memory cell in the profundity aspect LSTM, the decoder can answer both visual and language data adaptively. While in the traditional encoder-decoder structure, the profundity aspect LSTM isn't thought of, so the picture highlight stays consistent. With this plan, the significant visual direction isn't confusedly brought into the LSTM with commotion yet is specifically impacted by the created token. Moreover, as far as engineering, our model seems to be like a stacked LSTM , as both embrace a various leveled information structure. By and by, our model considers the created token and visual data straightforwardly in the profundity aspect LSTM, which keeps the model from veering off from the exact picture data. Moreover, we utilize the innate usefulness of the memory cell in the LSTM to adaptively incorporate the visual data. The memory cells work with data capacity and effectively code the conveyed input inside a solitary cell. Computational intricacy: The Recall Network works effectively with an update intricacy of O(W) per time step, where W is the quantity of boundaries. presents the computational subtleties for three distinct decoder units. Model (a) compares to the customary encoder-decoder system, where the picture portrayal is infused just a single time at the underlying advance. Model (b) relates to the model that takes additional data sources. Model (c) shows the computational subtleties of our Recall Network. Accept that the word implanting is a P-length vector, while the picture portrayal size and RNN stowed away unit size are both Q. Note that our Recall Network share boundaries along the transient aspect and the profundity aspect LSTM. Subsequently, the boundaries to be refreshed in our model are by and large equivalent to those in the traditional LSTM:

$W = 4\times(P \times Q+Q\times Q+Q)$.

Conversely, the quantity of boundaries in the model, which connects the

picture highlights as additional information sources , is $4\times((P +Q)\times Q+Q\times Q+Q)$. Assume $P =Q = 512$, our model contains around 1 million boundaries not exactly the model. Consequently, our model contains less teachable boundaries. One lack is that our model might require longer preparation time because of the calculations happening in two aspects.

4.4.1.3 Different Vision Models

For full and thick picture inscribing, we utilize different vision models. Full Image Captioning: In full picture subtitling, a sentence is created to portray the general substance of the full picture. We utilize a Resnet that is pre-prepared on ImageNet to introduce the CNN encoder. Thick Captioning: notwithstanding the full picture subtitling, we likewise explore our Recall Network for the thick inscribing task. Thick subtitling requires the model to not just identify the notable locales in a picture yet in addition to produce a depiction for every district,

which makes the portrayal explicit for every area of consideration. It achieves the restriction and subtitling undertakings mutually. We utilize the FCLN model in to acquire the area directions and locale highlights in a single picture. In the FCLN, a limitation layer is embedded between the convolutional layer and the completely associated layer of the VGG-16 network. The limitation layer gets an info tensor from the last convolutional layer of the VGG-16 organization, distinguishes the spatial district of the striking articles, and concentrates the fixed-length highlights for every area. The district recommendations are relapsed from a bunch of interpretation invariant anchors. Given an anchor box (xa, ya,wa,ha), the result area could be introduced by the middle (x, y) and shape (w,h) as follows:

x = xa +txwa,

y = ya +tyha,

w = wa exp(tw),

h = ha exp(th).

The (tx, ty, tw, th) are the parameters the model needs to learn. To generate a fixed-length feature for each region, and enable the error propagation through the coordinates, a bilinear interpolation is exploited as:

$$V_{C,i,j} = \sum_{i'=1}^{W} \sum_{j'=1}^{H} U_{C,i',j'} k\left(i' - x_{i,j}\right) k\left(j' - y_{i,j}\right),$$

k(d) = max(0,1|d|).

UC,i, j is the feature map in the shape of which is generated by the last convolutional layer of the VGG-16 network. The feature, VC,i, j , is then fed into a recognition network composed of two fully connected layers, then the final CNN features are fed into our language model. Tests In this work, we propose an original model, the Recall Network, for picture inscription age. The model can review the visual data ceaselessly and import the picture portrayal through the profundity aspect LSTM. To show the proficiency of our model, tests are performed on both full picture subtitling and thick inscribing. The expounded correlation incorporates the Recall Network versus the regular encoder decoder model, the Recall Network with Reinforcement Learning versus the traditional encoder-decoder model with Reinforcement Learning, the Recall Network versus the stacked LSTM, an examination with the best in class techniques on full picture subtitling disconnected and on the web, and the Recall Network versus

the standard techniques on thick subtitling.

4.4.1.4 Assessment Metrics

To assess whether the created portrayal is great, objective assessment measurements are used. For full picture subtitling, our trial results are accounted for utilizing the MSCOCO inscription assessment tool1, including the BLEU , METEOR , CIDEr and ROUGE-L measurements. BLEU is a well known machine interpretation metric that investigations the co-events of n-grams between the competitor and reference sentences, which has great execution for corpus-level examinations. METEOR is determined by producing an arrangement between the words in the up-and-comer and reference sentences, with the point of 1:1 correspondence. ROUGE is a bunch of assessment measurements intended to assess text rundown calculations. The CIDEr metric estimates the agreement in the picture inscriptions by playing out a Term Frequency Inverse Document Frequency (TF-IDF) weighting for every n-gram. We utilize the CIDEr metric to pick the models in the approving stage. Thick subtitling is a mix of article recognition and picture inscribing. Following , the mean Average Precision (AP) across a scope of edges for both confinement and language exactness is taken advantage of as the assessment metric. For limitation, we use Intersection over Union (IoU) edges of 0.3, 0.4, 0.5, 0.6 and 0.7. For the picture portrayal, we use METEOR edges of 0, 0.05, 0.1, 0.15, 0.2 and 0.25. Note that there is just a single depiction commented on inside a locale. We take on the METEOR since this measurement was viewed as exceptionally corresponded with human judgment in settings with a low number of references . We measure the normal accuracy across all the pairwise settings of these limits and report the mean AP.

4.4.2 Preparing Details

Pre-handling: For the MSCOCO dataset, we map every one of the words that happen under multiple times to the exceptional ⟨UNK⟩ token. For the Visual Genome dataset, we map words that happen under multiple times to the unique ⟨UNK⟩ token. We dispose of the comments with in excess of 10 words, and the pictures that have less than at least 20 than 50 explanations. We even union the intensely covering encloses to a solitary box. Execution subtleties: The full picture inscribing model is carried out utilizing PyTorch. The ResNet101 network, which is pretrained on the ImageNet dataset, is used as the picture encoder, and no finetuning is directed. We don't rescale or trim the pictures. We encode the full picture with the last convolutional layer and afterward apply normal pooling, which brings about a 2048-d vector. In the RNN decoder, both the RNN encoding size and word inserting size are set as 512. The dropout rate is set to 0.5 tentatively. We stop the preparation after 30 ages. The thick inscribing model is executed utilizing Torch. We utilize the FCLN as the locale encoder. The RNN size and word inserting size are set as 512. Following , we utilize five misfortunes

altogether. A load of 1.0 is alloted to the cross-entropy misfortune, while 0.1 is allocated as the load for other four misfortunes, remembering the double calculated misfortunes for district certainty for the limitation organization and acknowledgment organization, and the smooth L1 misfortune for the area position. At first, we utilize a learning pace of 1×104 for the language model, and a learning pace of 1×106 for tweaking the CNN. The Adam calculation is used for stochastic advancement. Our tests run on a NVIDIA TITAN X GPU. We start calibrating the CNN after 100,000 emphasess, and quit preparing after 600,000 cycles.

4.4.2.1 Investigates Full Images

For the full picture inscribing, we present the disconnected assessment, incorporating a correlation with baselines and best in class techniques, on similar information split as in [81, 101, 102], and present the internet based assessment results on the MSCOCO test server. The Resnet network is used as the encoder.

The Recall Network versus the Conventional Encoder-Decoder Model

In the first place, we contrast our Recall Network and the pattern, traditional encoder-decoder model (the CEM), to see the adequacy of our model. The CEM is a variety of the NIC model where we reimplement it with the ResNet as the picture encoder. The correlation results are introduced in. As the shaft search can support the presentation by roughly a couple of percent, we present the outcomes with eager example as well as pillar search with size 2. Notwithstanding the essential outcomes, we likewise show the presentation when prepared with the support learning (RL) strategy , which can help the trial results over the fundamental models. From, we can see that our Recall Network beats the correlation model in every one of the measurements. It is significant that our Recall Network beats the customary encoder-decoder model by a huge degree in the CIDEr metric, where the exhibition is supported from 94.50 (99.09 with the bar search) to 98.31 (101.55 with the shaft search). Further utilizing the Reinforcement Learning preparing strategy our Recall Network accomplishes 103.70 in the CIDEr metric. With RL preparing, our organization outflanks the correlation model in every one of the measurements aside from METEOR. It is hypothesized that the METEOR results might be affected on the grounds that we utilize the CIDEr score not METEOR as remuneration in the RL preparing.

Chapter 5

RESULTS AND DISCUSSION

This chapter gives a brief description about the input used and the expected output of the system.

5.1 Object detection

The Figure represents results of object detection. Which is detecting the multiple objects in one pram

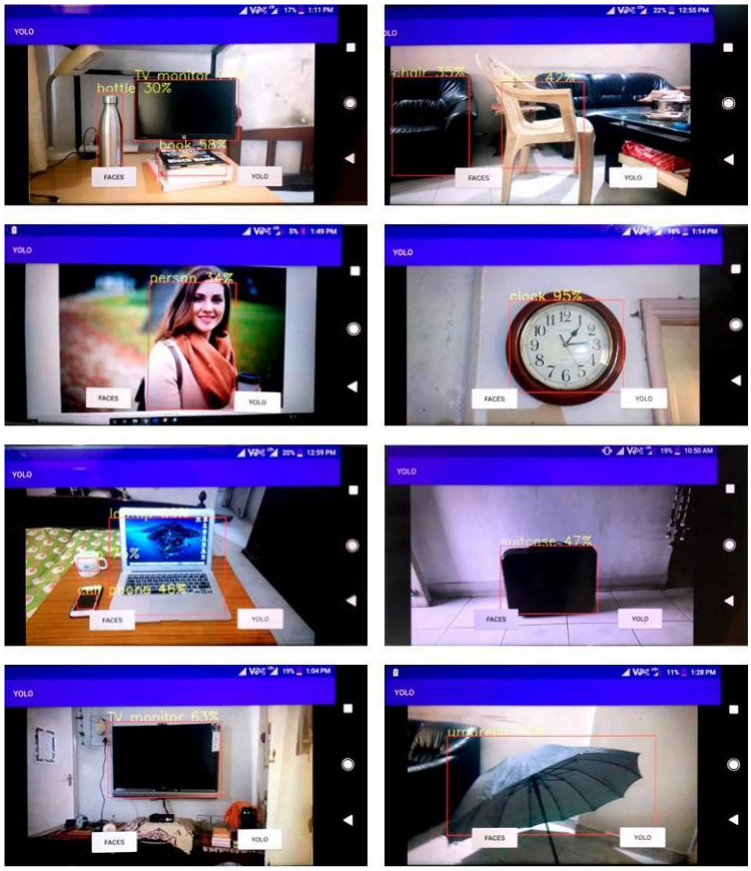


Figure 5.1: Results of object detection

The figure bellow is showing a person with his gender,age,and his emotion the above image previously not trained so its showing unknown with other details like age,gender,emotion except his name.

Figure 5.2: Results of object detection

The bellow figure detection multiple person in single frame

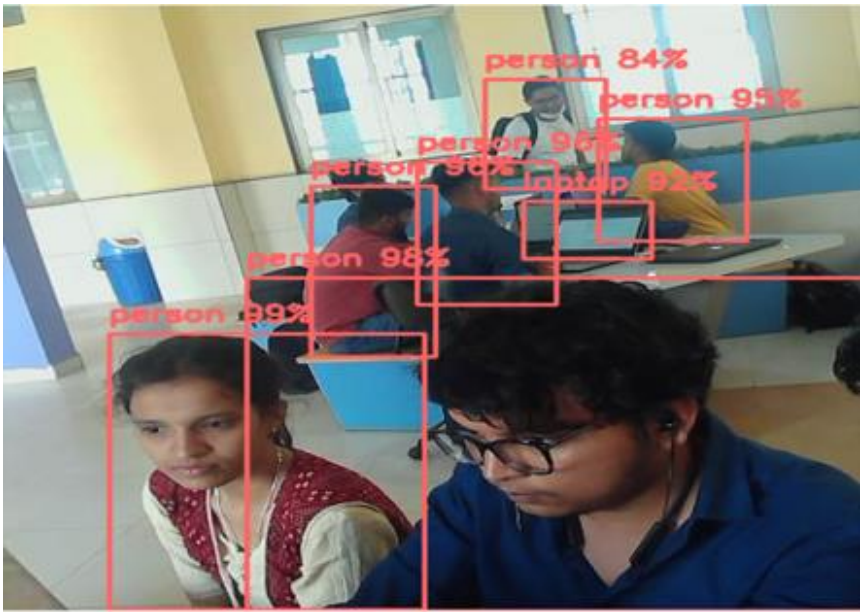


Figure 5.3: Results of object detection

5.2 Emotion detection

The bellow image is trained so it is showing all the details like name,gender,age,emotion of the person.

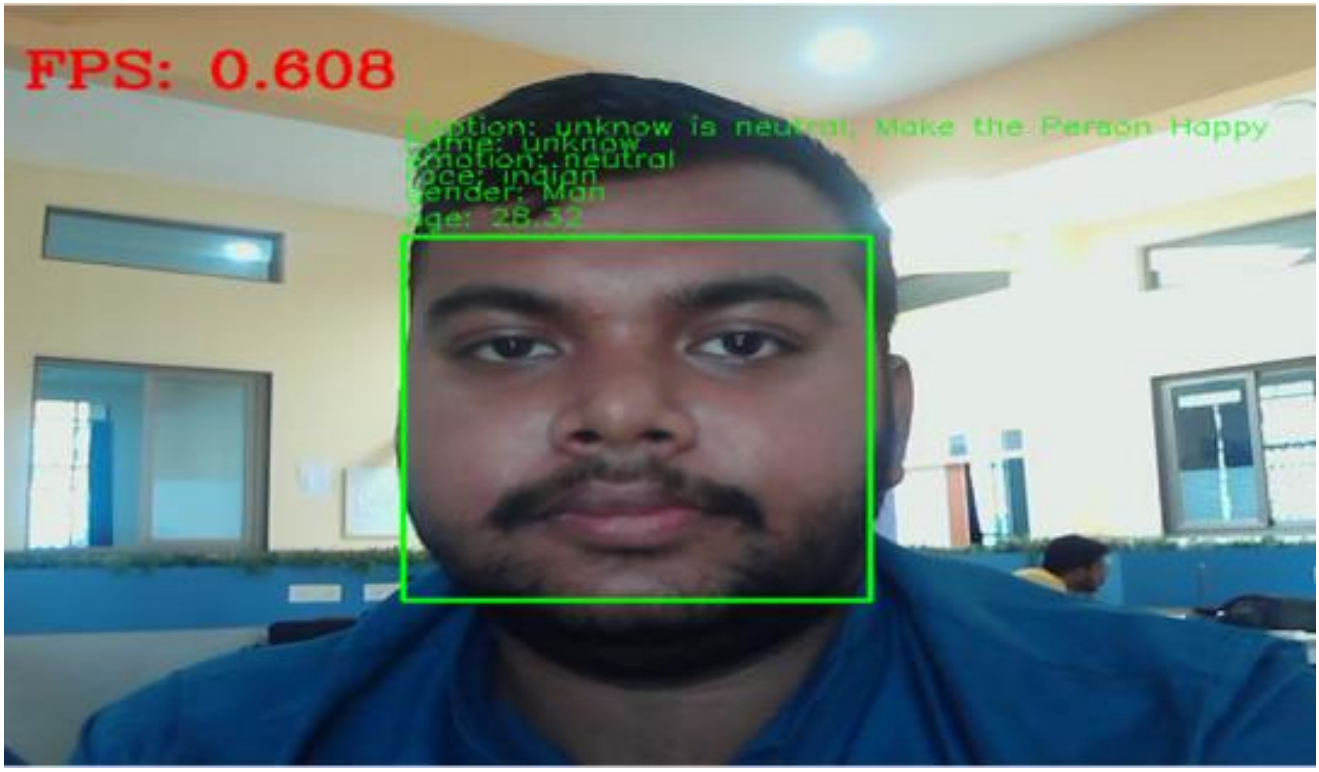


Figure 5.4: Results of emotion detection

The Figure gives the test set of images of all four fruits to test the system inorder to recognise the fruits, around 35 images of four fruits are given as test set of images.

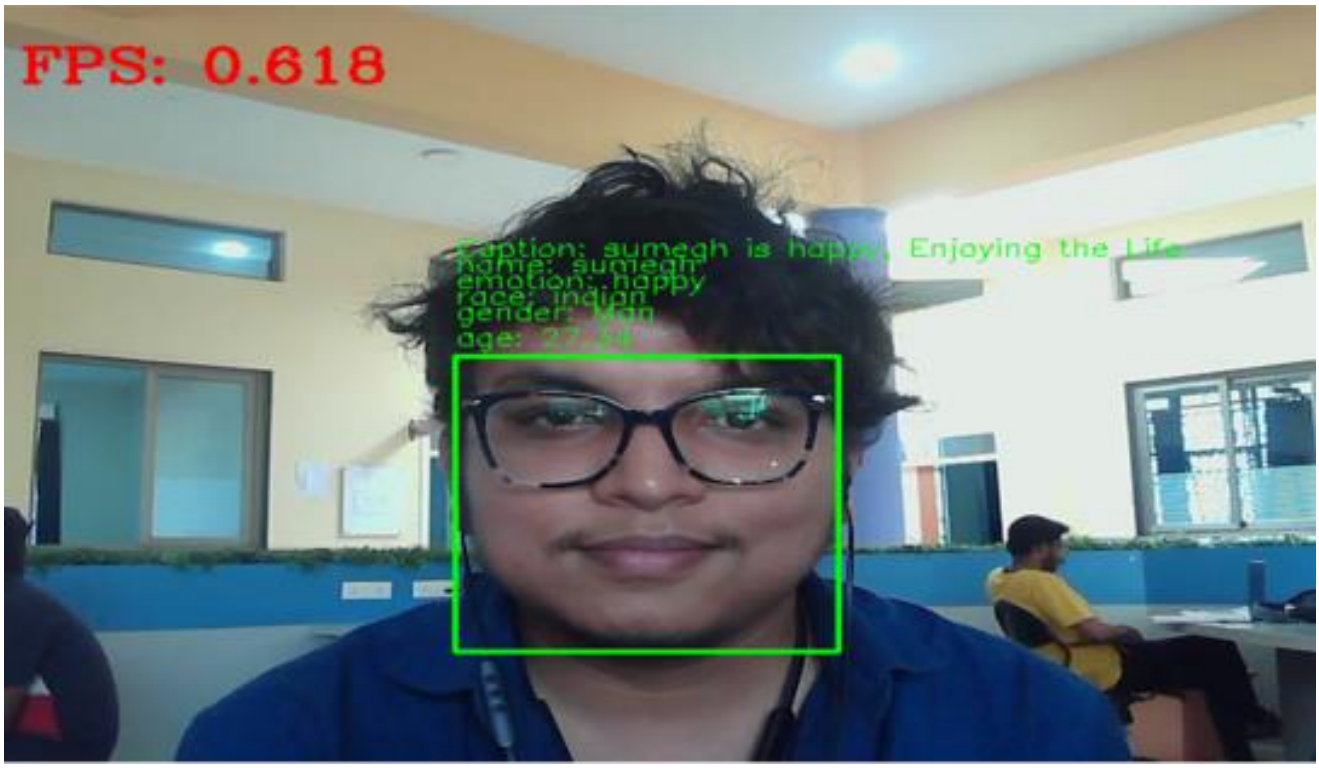


Figure 5.5: Results of emotion detectiom

Chapter 6

CONCLUSION

One of the fundamental issues including object location is object grouping and item localisation inside a scene. Utilization of profound brain networks has helped in tending to the subject of item location. Notwithstanding, conveying such strategies on cell phones requires high computational and memory assets. Subsequently involving little profound brain network structures for object discovery like Tiny Consequences be damned are giving great outcomes and show that they can be utilized for constant article recognition utilizing cell phones which can help the outwardly tested. As high exactness is expected for face acknowledgment, we have effectively involved FaceNet engineering for face acknowledgment on an Android gadget. Profound Learning is as yet arising out as an enormous innovation which will assist people with to not just perceive faces in pictures or recordings or discourse or sound interpretations however to give people another species who might convey comparable cerebrums as human does. It is only the essential and essential thought that I have examined and conveyed my examination on. The common models of brain networks are without a doubt the foundation of such effective future profound learning holds. The models like CNN, Deep CNN and other popular preparation calculations and different noising or de noising encoders and decoders expounds the profound learning methods and makes sense of that profound learning has not demonstrated and exhibited promising outcomes in a considerable lot of its applications yet additionally has the promising future with holding achievement and more extensive viewpoints. There are as yet many necessities that are still to be created utilizing profound learning such making do of face acknowledgment exactness rate, discourse identification and interpretation, better preparation of the organizations, simpler preparation of machines driving us to more powerful and hypothetical methodologies in the field of profound learning. We want to observe better component extraction at each layer of the brain network as we have observed that the essential back proliferation calculation was practically unequipped for doing the errand that grid type of back engendering calculation does as late investigations and works have shown that there is compelling reason need to re train the entire model or organization rather one can just refresh it utilizing new information and data. It is additionally important to create and investigate the procedures that are all the more remarkable and has the discriminative methodology towards streamlining. Albeit the current learning approaches of profound learning appears to function admirably observationally for some errands yet they actually fizzled for a few different undertakings, for example, robot protecting the pool suffocated himself in

the pool in light of absence of information and preparing. Separating highlights from each layer of the organization or even from each contribution of the organization can give profound learning more modest size models and calculations. We likewise need to think of more versatile models that involves the equal methodology in preparing profound models or organizations which can prompt enhancement of different calculations. The current best answer for upgrade these models is the utilization of Graphical Processing Units (GPUs) yet on the off chance that we decrease the quantity of GPUs or on the other hand on the off chance that we utilize just single machine, this act of utilizing GPUs isn't common sense enough for huge datasets and data. Making profound learning procedures that are versatile and equal learning calculation that would require some investment to prepare should be created.

D_20

ORIGINALITY REPORT

14%	12%	2%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	scholarworks.bridgeport.edu Internet Source	6%
2	hdl.handle.net Internet Source	3%
3	opus.lib.uts.edu.au Internet Source	3%
4	Lingxiang Wu, Min Xu, Jinqiao Wang, Stuart Perry. "Recall What You See Continually Using GridLSTM in Image Captioning", IEEE Transactions on Multimedia, 2020 Publication	2%
5	www.ijres.org Internet Source	<1%
6	dspace.vutbr.cz Internet Source	<1%
7	acikerisim.karatay.edu.tr:8080 Internet Source	<1%
8	i2labs.co Internet Source	<1%

9	idswater.com Internet Source	<1%
10	careerkarma.com Internet Source	<1%
11	wiki.openbravo.com Internet Source	<1%
12	www.mdpi.com Internet Source	<1%

Exclude quotes	On	Exclude matches	< 10 words
Exclude bibliography	On		

References

[1] Marcus Fontoura Alexander Shraer, Maxim Gurevich and Vanja Josifovski. "Top-k Publish-Subscribe for Social Annotation of New". *Proceedings of the VLDB Endowment*, pages 6(6):385–396, 26th August 2013.

[2] From Lucene Apache. Lucene search engine. http://lucene.apache.org.

[3] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, 2016, The MIT Press, Cambridge, MA, USA.

[4] Bernard, Marr. "Artificial Intelligence: What's the Difference between Deep Learning and Reinforcement Learning", 2018, Forbes, USA.

[5] Karn, Ujjwal. "A Quick Introduction to Neural Networks", 2016, the Data Science Blog.

[6] Graupe, Daniel, "Principle of artificial Neural networks", 2013, World Scientific Publishing Co Pte Ltd.

[7] Zhang, Shunlu. Gunupudi Pavan. Zhang Qi-Jun. "Parallel Back-Propagation Neural Network Training Technique Using CUDA on Multiple GPUs", 2015 IEEE MTT-S International Conference on Numerical Electromagnetic and Multi physics Modeling and Optimization (NEMO).

[8] Sierra-Canto, F. Madera-Ramrez and V. Uc-Cetina, Parallel training of a back-propagation neural network using CUDA, 2010 Ninth Int. Conf. on Machine Learning and Applications, Washington, pp. 307-312, Dec.2010.

[9] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Back propagation applied to handwritten zip code recognition. Neural Computation, 1(4):541–551, Dec. 1989. 2.

[10] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back propagating errors. Nature, 1986. 2, 4.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. CoRR, abs/1409.4842, 2014. 2, 4, 5, 6, 9.

[12] Florian Schroff, Dmitry Kalenichenko, James Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering", in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2015.

[13] Y. Sun, X. Wang, and X. Tang. Deep learning faces representation by joint identification verification. CoRR, abs/1406.4773, 2014. 1, 2, 3.

[14] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In NIPS. MIT Press, 2006. 2, 3.

[15] Y. Sun, X. Wang, and X. Tang. Deeply learned face representations are sparse, selective, and robust. CoRR, abs/1412.1265, 2014. 1, 2, 5, 8.

[16] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deep face: Closing the gap to human-level performance in face verification. In IEEE Conf. on CVPR, 2014. 1, 2, 5, 8.

[17] Sombir Singh Bisht et al Int. Journal of Engineering Research and Applications

[18] J.P.W. Pluim; J.M. Fitzpatrick, "Image Registration", IEEE Transactions on Medical Imaging (Volume: 22, Issue: 11, Nov. 2003).

[19] Adrian Rosebrock, "Face Alignment with OpenCV and Python", 2017, PyImageSearch.

[20] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "You Only Look Once, Unified, Real time Object Detection", In IEEE Conf. on CVPR, 2016.