

# **MACHINE-GENERATED CAPTIONS FOR IMAGES USING DEEP LEARNING**

Submitted in partial fulfillment of the  
requirements for the award of  
Bachelor of Engineering degree in Computer Science and Engineering  
By

**KOSANAM SRINIVAS (REG.NO - 39110527)**

**KARWAN VISHWESHWAR (REG.NO - 39110469)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SCHOOL OF COMPUTING**

## **SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE  
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,  
CHENNAI - 600119**

**APRIL - 2023**



# **SATHYABAMA**

INSTITUTE OF SCIENCE AND  
TECHNOLOGY

**(DEEMED TO BE UNIVERSITY)**

Accredited with —AI grade by NAAC Jeppiaar  
Nagar, Rajiv Gandhi Salai, Chennai – 600 119  
[www.sathyabama.ac.in](http://www.sathyabama.ac.in)



## **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

### **BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **KOSANAM SRINIVAS (REG.NO - 39110527)** and **KARWAN VISHWESHWAR (REG.NO - 39110469)** who carried out the Project Phase-1 entitled “**MACHINE-GENERATED CAPTIONS FOR IMAGES USING DEEP LEARNING**” under my supervision from January 2023 to April 2023.

**Internal Guide**

**MS.C.A. DAPHINE DESONA CLEMENCY, M.E.**

**Head of the Department**

**Dr. L. LAKSHMANAN, M.E., Ph.D.**



**Submitted for Viva voce Examination held on 19.4.2023**

**Internal Examiner**

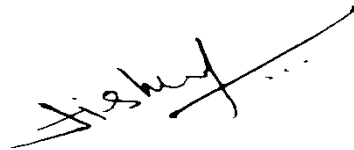
**External Examiner**

## DECLARATION

I, **KOSANAM SRINIVAS (REG.NO - 39110527)**, hereby declare that the Project Phase-1 Report entitled “**MACHINE-GENERATED CAPTIONS FOR IMAGES USING DEEP LEARNING**” done by me under the guidance of **MS.C.A. DAPHINE DESONA CLEMENCY, M.E.** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **COMPUTER SCIENCE AND ENGINEERING**.

**DATE: 19.4.2023**

**PLACE: Chennai**

A handwritten signature in black ink, appearing to read 'Kosanam Srinivas', written diagonally across the page.

**SIGNATURE OF THE CANDIDATE**

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph. D, Dean**, School of Computing, and **Dr. L. Lakshmanan M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Ms.C.A. Daphine Desona Clemency, M.E.**, for her valuable guidance, suggestions, and constant encouragement paved way for the successful completion of my phase-1 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

## ABSTRACT

The image caption generator's main aim is to generate an English sentence or caption for an image automatically. The main challenge of the system is to generate correct captions for the given image. This paper proposes an image caption generator that will accept an image as an input and generate an English sentence as output by labeling the image's content using two optimization techniques such as beam search and greedy search. The system takes the pre-trained deep learning Convolutional Neural Network (CNN) architecture VGG16 model for learning the image features, uses Long Short-Term Memory (LSTM) for learning the text features, and combines the image's result with an LSTM to generate a caption for the image. We use the LSTM model to generate text or sentences or captions for the given input images. This model was developed to build an image caption generator by implementing the Convolutional Neural Network with Long Short-Term Memory. The pre-trained VGG16 is used to extract features from the given image. LSTM works as a decoder to generate sentences or captions for the images. This model is trained so that if the input image is given to the model, it will generate captions or sentences describing the image. The system is evaluated with bleu scores to evaluate the model's efficiency. We discuss Keras library, NumPy and Jupyter notebooks for the making of this project. We also discuss about Flickr dataset and CNN used for image classification

**Keywords:** Generate captions, Deep learning techniques, and concepts of image captioning.

## **TABLE OF CONTENTS**

<b>Chapter No</b>	<b>TITLE</b>	<b>Page No.</b>
	<b>ABSTRACT</b>	<b>v</b>
	<b>LIST OF FIGURES</b>	<b>ix</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 Problem Statement	4
	1.2 Motivation	5
<b>2</b>	<b>LITERATURE SURVEY</b>	<b>7</b>
	2.1 Image Captioning Methods	7
	2.2 Inferences From Literature Survey	10
	2.3 Open Problems In Existing System	11
	2.4 Existing System	12
	2.5 Other Deep Learning-Based Image Captioning	12
	2.5.1 Dense Captioning	12
	2.5.2 Captions For The Whole Scene	12
	2.6 LSTM vs. Others	14
	2.7 Mostly Referred Research Papers	15
<b>3</b>	<b>REQUIREMENTS ANALYSIS</b>	<b>17</b>
	3.1 Feasibility Studies/Risk Analysis of the Project	17
	3.1.1 Economical Feasibility	17
	3.1.2 Technical Feasibility	17
	3.1.3 Social Feasibility	17

3.2	Software Requirements Specification Document	18
3.3	Language Specification	19
3.4	Application Of Python	0
3.5	Different Libraries In Python	21
3.6	Features Of Python	22
3.7	Software Installation For Image Caption Generation Projects	22
3.7.1	Installing Python	22
3.7.2	Installing PyCharm	23
3.8	System Use Cases	25
3.9	Import All the Required Package	27
<b>4</b>	<b>DESCRIPTION OF PROPOSED SYSTEM</b>	<b>28</b>
4.1	Proposed System	28
4.2	Selected Methodology or process model	29
4.2.1	CNN Algorithm	29
4.2.2	Long Short-Term Memory	37
4.3	Architecture / Overall Design of Proposed System	37
4.4	Description of Software for Implementation and Testing plan of the Proposed Model/System	38
4.4.1	Input Design	38
4.4.2	Output Design	39
4.4.3	Plan of Proposed System	39
4.4	Project Management Plan	41
4.4.1	Purpose	42
4..4.2	Scope	42
<b>5</b>	<b>IMPLEMENTATION DETAILS</b>	<b>43</b>
5.1	Development And Deployment Setup	43

5.1.1	Modules	43
5.1.2	Building The model	43
5.2	Algorithms	44
5.3	System Testing	45
5.4	Steps For Executing The Projects	46
5.5	Activity Diagram	46
5.6	Er Diagram	47
<b>6</b>	<b>RESULTS AND DISCUSSION</b>	<b>48</b>
6.1	Caption Generation For Image	48
6.2	Result Of Image Caption Generation	49
<b>7</b>	<b>CONCLUSION</b>	<b>50</b>
7.1	Conclusion	50
7.2	Future work	50
7.3	Research Issues	50
7.4	Implementation Issues	50
	<b>REFERENCES</b>	<b>51</b>
	<b>APPENDIX</b>	<b>53</b>
	<b>A. SOURCE CODE</b>	<b>53</b>
	<b>B. SCREENSHOTS</b>	<b>56</b>
	<b>C. RESEARCH PAPER</b>	<b>57</b>



## **LIST OF FIGURES**

<b>Figure No</b>	<b>FIGURE NAME</b>	<b>Page No.</b>
1.1	An Overall Taxonomy of Deep Learning-Based Image Captioning.	6
2.1	A Block Diagram of Simple Encoder-Decoder Architecture-Based Image Captioning	12
2.2	A block diagram of a compositional network-based captioning	14
3.1	A block diagram of a Python installation	22
3.2	A block diagram of a PyCharm installation	23
3.3	A block diagram of a create new project on PyCharm	24
3.4	Login Page and Registration use case diagram	26
3.5	A block diagram of a Class Diagram	26
4.1	Real-Time Image Captioning on an Embedded System using a light-weight Deep Learning	28
4.2	Convolution Layer	32
4.3	Fully Connected Layer	36
4.4	General Architecture Of The CNN-LSTM Model	38
4.5	Image caption Generator Using Deep Learning Architecture	38
4.6	Plan Of Proposed Model	40
4.7	Project Management Plan	41
5.1	Workflow of Project	47
5.2	Relationship Model Diagram	47
6.1	Process Of Application	48
6.2	Result of Image caption generation	49

# CHAPTER 1

## INTRODUCTION

Every day, we encounter a large number of images from various sources such as the internet, news articles, document diagrams, and advertisements. These sources contain images that viewers would have to interpret themselves. Most images do not have a description, but humans can largely understand them without their detailed captions.

. Human brain is good enough to understand most of the images even when they do not have any descriptions assigned to them. On the other hand, for machines it is a difficult task to caption an image. Forming natural English captions to any image automatically is a challenging task. But, once it is done, it could have a significant impact in our society. An image caption should not only capture the objects/elements in them, but it should also be able to relate them with respect to their activities etc.

Automatically generating captions could help the visually impaired to understand and feel the situation in an image. It helps in the enormous indexing of images done by Google LLC and others to create an image search engine. Social media platforms can use it for tagging and image captioning purposes. Military around the world might use them for reconnaissance purposes.

Traditional machine learning methods use Local Binary Patterns (LBP) etc., are used. The features extracted are then passed on to any classification algorithm like Support Vector Machines (SVM) to classify the images. Such a model depends upon hand-selected features and hence their use is not feasible for large datasets, especially with large features. Manually designing features for a complex task can require enormous amounts of human labor and time.

Modern image captioning methods use deep learning techniques does not require us to specify all the features that we want. They automatically capture the features and hence such methods are highly scalable. Deep learning solves the problem of feature extraction and selection by representing the representation in the form in simpler representations in multiple layers (hence the name Deep Learning).

Convolution Neural Network (CNN) is widely used for automatic feature extraction from images which is generally followed by Long Short-Term Memory (LSTM) to predict the captions one by one in sequence of words.

Convolution Neural Networks (CNN), unlike Multi-Layer Perceptron (MLP) which are also known as vanilla neural networks do not have a fully-connected property. In MLP, the fully-connected property implies that each neuron is connected to every other neuron which makes the model prone to overfitting.

However, a machine needs to interpret some form of image captions if humans need automatic image captions from it. Image captioning is important for many reasons. Captions for every image on the internet can lead to faster and descriptively accurate image searches and indexing.

Ever since researchers started working on object recognition in images, it became clear that only providing the names of the objects recognized does not make such a good impression as a full human-like description. As long as machines do not think, talk, and behave like humans, natural language descriptions will remain a challenge to be solved.

Image captioning has various applications in various fields such as biomedicine, commerce, web searching, and military, etc. Social media like Instagram, Facebook, etc. can generate captions automatically from images.

Generating a caption for a given image is a challenging problem in the deep learning domain. In this article, we will use different techniques of computer vision and NLP to recognize the context of an image and describe them in a natural language like English. We will build a working model of the image caption generator by using CNN (Convolutional Neural Networks) and LSTM (Long short-term memory) units.

- You saw an image and your brain can easily tell what the image is about, but can a computer tell what the image is representing? Computer vision researchers worked on this a lot and they considered it impossible until now! With the advancement in Deep learning techniques, availability of huge datasets and computer power, we can build models that can generate captions for an image.
- This is what we are going to implement in this Python based project where we will use deep learning techniques of Convolutional Neural Networks and a type of Recurrent Neural Network (LSTM) together.

- Image caption generator is a task that involves computer vision and natural language processing concepts to recognize the context of an image and describe them in a natural language like English.

The people communicate through language, whether written or spoken. They often use this language to describe the visual world around them. Images, signs are another way of communication and understanding for the physically challenged people. The generation of descriptions from the image automatically in proper sentences is a very difficult and challenging task [1], but it can help and have a great impact on visually impaired people for better understanding of the description of images on the web.

A good description of an image is often said for 'Visualizing a picture in the mind'. The creation of an image in mind can play a significant role in sentence generation. Also, human can describe the image after having a quick glance at it. The progress in achieving complex goals of human recognition will be done after studying existing natural image descriptions. This task of automatically generating captions and describing the image is significantly harder than image classification and object recognition.

The description of an image must involve not only the objects in the image, but also relation between the objects with their attributes and activities shown in images. Most of the work done in visual recognition previously has concentrated to label images with already fixed classes or categories leading to the large progress in this field. Eventually, vocabularies of visual concepts which are closed, makes a suitable and simple model for assumption.

These concepts appear widely limited after comparing them with the tremendous amount of thinking power which human possesses. However, the natural language like English should be used to express above semantic knowledge, that is, for visual understanding language model is necessary. Fig. 1. Model based on Neural Networks In order to generate a description from an image, most of the previous attempts have suggested combining all the current solutions of the above problem. Whereas, we will be designing a single model which takes an image as an input and is trained for producing a sequence of words where each word belongs to the dictionary that describes the image suitably.

The relation between visual importance and descriptions moves to the text summarization problem in natural language processing (NLP) [18]. The important

goal of text summarization is selecting or generating an abstract for document.

In the problem of image captioning, for any image we would like to generate a caption that will describe various features of that image. This paper proposes a model capable of generating novel descriptions from images. For this task, we have used the Flickr 8k dataset consisting of 8000 images and five descriptions per image. This illustrates the dataset structure in which an image is having five natural language captions. In this work, we are using CNN as well as RNN. Pre-trained Convolutional Neural Network (CNN) is used for the image classification task.

This network acts as an image encoder. The last hidden layer is used as an input to Recurrent Neural Network (RNN). This network is a decoder which generates sentences. Sometimes, the generated sentence seems to lose track or predict wrong sentence than that of the original image content.

This work related to the methods of third category of caption generation. In this approach, a neural model is designed which generates descriptions for image in natural language. CNN is used as image encoder. Firstly, pre-training is done for image classification task and then the RNN decoder uses this last hidden layer as input to generate the sentence.

Convolution networks uses the mathematical operation convolution instead of matrix multiplication which the usual neural networks uses.

Long Short-Term Memory (LSTM) cannot just process individual images, but can also process sequence of images or videos. It is a type of RNN model. An RNN is a type of neural network which is repeated again and again and the newer repetitions are only dependent on the previous one computation enabling the processing of a sequence of data. But, RNN poses the problem of vanishing gradient for multi layered model. It can make the model harder to train. To prevent the problem to vanishing gradients, we use Long Short-Term Memory (LSTM) which is a modified version of RNN.

We merge both CNN and LSTM model to produce results using SoftMax. SoftMax takes in a vector of numbers and converts them to probabilities which are then used for image generating results.

## **1.1 PROBLEM STATEMENT**

Based on the content of an Image, this application will generate a caption for any natural image. Such an application might help blind people to see the world full with images. This model automatically generates natural language captions which

then can be utilized for indexing and searching of images, tagging in social media, helping the visually impaired etc. The uses of such an application is immense. In this paper, we will build such an application using Convolution Neural Networks (CNN) for feature extraction and Long Short-Term Memory (LSTM) for generating the captions.

## **1.2 MOTIVATION**

Generating captions for images is a vital task relevant to the area of both Computer Vision and Natural Language Processing. Mimicking the human ability to provide descriptions for images by a machine is itself a remarkable step along the line of Artificial Intelligence.

The main challenge of this task is to capture how objects relate to each other in the image and to express them in a natural language (like English). Traditionally, computer systems have been using pre-defined templates for generating text descriptions for images.

However, <sup>1</sup> this approach does not provide sufficient variety required for generating lexically rich text descriptions. This shortcoming has been suppressed with the increased efficiency of neural networks. Many states of art models use neural networks for generating captions by taking images as input and predicting the next lexical unit in the output sentence.

## **1.3 IMAGE CAPTIONING**

**Process:** - Image Captioning is the process of generating a textual description of an image. It uses both Natural Language Processing and Computer Vision to generate the captions. Image captioning is a popular research area of Artificial Intelligence (AI) that deals with image understanding and a language description for that image. Image understanding needs to detect and recognize objects. It also needs to understand scene type or location, object properties and their interactions.

Generating well-formed sentences requires both syntactic and semantic understanding of the language. Understanding an image largely depends on obtaining image features. For example, they can be used for automatic image indexing.

Image indexing is important for Content-Based Image Retrieval (CBIR) and therefore, it can be applied to many areas, including biomedicine, commerce, the military, education, digital libraries, and web searching. Social media platforms such

as Facebook and Twitter can directly generate descriptions from images. The descriptions can include where we are (e.g., beach, cafe), what we wear and importantly what we are doing there.

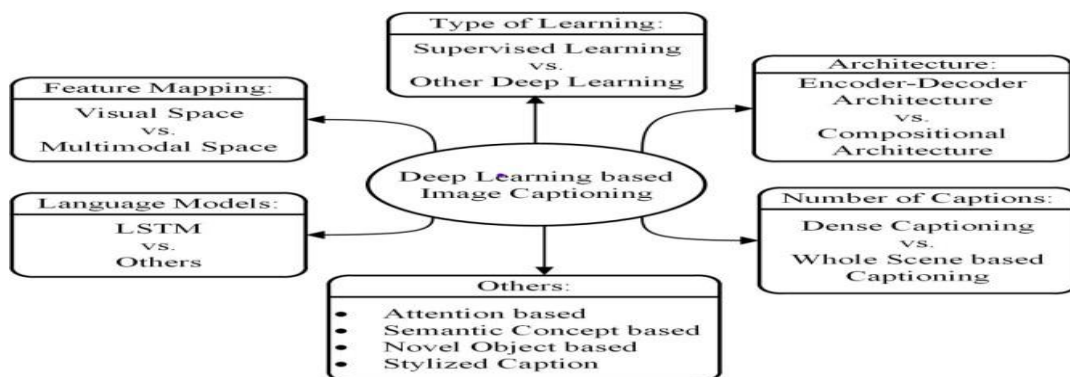
**Techniques:** - The techniques used for this purpose can be broadly divided into two categories: (1) Traditional machine learning-based techniques and (2) Deep machine learning-based techniques. In traditional machine learning, hand-crafted features such as Local Binary Patterns (LBP) [17], Scale-Invariant Feature Transform (SIFT) [8], the Histogram of Oriented Gradients (HOG) [7], and a combination of such features are widely used. In these techniques, features are extracted from input data.

They are then passed to a classifier such as Support Vector Machines (SVM) [15] in order to classify an object. Since hand-crafted features are task-specific, extracting features from a large and diverse set of data is not feasible. Moreover, real-world data such as images and videos are complex and have different semantic interpretations.

On the other hand, in deep machine learning-based techniques, features are learned automatically from training data and they can handle a large and diverse set of images and videos.

For example, Convolutional Neural Networks (CNN) [9] are widely used for feature learning, and a classifier such as SoftMax is used for classification.

CNN is generally followed by Recurrent Neural Networks (RNN) or Long Short-Term Memory Networks (LSTM) in order to generate 10 captions. Deep learning algorithms can handle the complexities and challenges of image captioning quite well.



**Fig.1.1 An Overall Taxonomy of Deep Learning-Based Image Captioning.**

## **CHAPTER 2**

### **LITERATURE SURVEY**

Image captioning has recently gathered a lot of attention specifically in the natural language domain. There is a pressing need for context-based natural language description of images, however, this may seem a bit farfetched but recent developments in fields like neural networks, computer vision, and natural language processing has paved a way for accurately describing images i.e., representing their visually grounded meaning. We are leveraging state-of-the-art techniques like Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and appropriate datasets of images and their human perceived description to achieve the same. We demonstrate that our alignment model produces results in retrieval experiments on datasets such as Flickr.

#### **2.1 IMAGE CAPTIONING METHODS**

There are various Image Captioning Techniques some are rarely used in present but it is necessary to take an overview of those technologies before proceeding ahead. The main categories of existing image captioning methods and include template-based image captioning, retrieval-based image captioning, and novel caption generation. Novel caption generation-based image caption methods mostly use visual space and deep machine learning-based techniques. Captions can also be generated from multimodal space. Deep learning-based image captioning methods can also be categorized into learning techniques: Supervised learning, Reinforcement learning, and Unsupervised learning. We group reinforcement learning and unsupervised learning into Other Deep Learning.

Usually, captions are generated for a whole scene in the image. However, captions can also be generated for different regions of an image (Dense captioning). Image captioning methods can use either simple Encoder-Decoder architecture or Compositional architecture. There are methods that use attention mechanisms, semantic concepts, and different styles in image descriptions. Some methods can also generate descriptions for unseen objects. We group them into one category as "Others". Most of the image captioning methods use LSTM as language model. However, there are a number of methods that use other language models such as CNN and RNN. Therefore, we include a language model-based category as "LSTM vs. Others".



## IMAGE CAPTION GENERATION USING DEEP LEARNING TECHNIQUE

**Abstract**—In Artificial Intelligence (AI), the contents of an image are generated automatically which involves computer vision and NLP (Natural Language Processing). The neural model, which is regenerative, is created. It depends on computer vision and machine translation. This model is used to generate natural sentences which eventually describe the image. This model consists of Convolutional Neural Network (CNN) as well as Recurrent Neural Network (RNN). The CNN is used for feature extraction from images and RNN is used for sentence generation. The model is trained in such a way that if the input image is given to the model it generates captions that nearly describe the image. The accuracy of model and smoothness or command of the language model learned from image descriptions is tested on different datasets. These experiments show that model is frequently giving accurate descriptions for an input image.

## A NEURAL IMAGE CAPTION GENERATOR.” COMPUTER VISION AND PATTERN RECOGNITION

Automatically describing the content of an image is a fundamental problem in artificial intelligence that connects computer vision and natural language processing. In this paper, we present a generative model based on a deep recurrent architecture that combines recent advances in computer vision and machine translation and that can be used to generate natural sentences describing an image. The model is trained to maximize the likelihood of the target description sentence given the training image. Experiments on several datasets show the accuracy of the model and the fluency of the language it learns solely from image descriptions. Our model is often quite accurate, which we verify both qualitatively and quantitatively. For instance, while the current state-of-the-art BLEU-1 score (the higher the better) on the Pascal dataset is 25, our approach yields 59, to be compared to human performance around 69. We also show BLEU-1 score improvements on Flickr30k, from 56 to 66, and on SBU, from 19 to 28. Lastly, on the newly released COCO dataset, we achieve a BLEU-4 of 27.7,

## OVERFEAT: INTEGRATED RECOGNITION, LOCALIZATION AND DETECTION USING CONVOLUTIONAL NETWORKS

We present an integrated framework for using Convolutional Networks for classification, localization, and detection. We show how a multiscale and sliding

window approach can be efficiently implemented within a ConvNet. We also introduce a novel deep-learning approach to localization by learning to predict object boundaries. Bounding boxes are then accumulated rather than suppressed in order to increase detection confidence. We show that different tasks can be learned simultaneously using a single shared network. This integrated framework is the winner of the localization task of the ImageNet Large Scale Visual Recognition Challenge 2013 (ILSVRC2013) and obtained very competitive results for the detection and classification tasks. In post-competition work, we establish a new state-of-the-art for the detection task. Finally, we release a feature extractor from our best model called OverFeat.

#### EVERY PICTURE TELLS A STORY: Generating Sentences From Images

Humans can prepare concise descriptions of pictures, focusing on what they find important. We demonstrate that automatic methods can do so too. We describe a system that can compute a score linking an image to a sentence. This score can be used to attach a descriptive sentence to a given image or to obtain images that illustrate a given sentence. The score is obtained by comparing an estimate of meaning obtained from the image to one obtained from the sentence. Each estimate of meaning comes from a discriminative procedure that is learned using data. We evaluate a novel dataset consisting of human-annotated images. While our underlying estimate of meaning is impoverished, it is sufficient to produce very good quantitative results, evaluated with a novel score that can account for synecdoche.

#### ATTENTION MECHANISM-BASED CNN FOR IMAGE CAPTURE GENERATED

Image capture generated is a hot research topic and can be applied in many computer vision fields, such as human-computer interaction, affective computing and so on. In this paper, we propose a novel end-to-end network with attention mechanism for automatic facial expression recognition. The new network architecture consists of four parts, i.e., the feature extraction module, the attention module, the reconstruction module, and the classification module. The LBP features extract image texture information and then catch the small movements of the faces, which can improve the network performance. Attention mechanism can make the neural network pay more attention to useful features. We combine LBP features and attention mechanism to enhance the attention model to obtain better results. In addition, we collected and labelled a new facial expression dataset of seven

expressions from 35 subjects aged from 20 to 25. For each subject, we captured both RGB images and depth images with a Microsoft Kinect sensor. For each image type, there are 245 image sequences, each of which contains 110 images, resulting in 26950 images in total. We apply the newly proposed method to our own dataset and four representative expression datasets, i.e., JAFFE, CK+, FER2013 and Oulu-CASIA. The experimental results demonstrate the feasibility and effectiveness of the proposed method.

## IMAGE CAPTURE GENERATED USING MULTI-DEEP CONVOLUTIONAL NEURAL NETWORK ENCODERS WITH SUPPORT VECTOR MACHINES

Although there have been many breakthroughs in the use of convolutional neural networks (CNN) for image classification, image capture generated (FER) in real-life is still a challenge in this research area. This paper proposes a method to leverage state-of-the-art multi-deep CNN encoders with support vector machines (SVM) to classify facial expressions.

We conducted experiments to show that combining features from multi-deep CNN is better than using a single-deep CNN model. As well as combining multiple CNN models, we show that using rules to remove noise images from the training dataset improves the performance of the FER system. The FER2013 dataset was used to evaluate the proposed approach, which achieved 73.78% accuracy.

The scope of this Generating a caption for a given image is a challenging problem in the deep learning domain.

## 2.2 INFERENCES FROM THE LITERATURE SURVEY

Image captioning has recently gathered a lot of attention specifically in the natural language domain. There is a pressing need for context-based natural language description of images, however, this may seem a bit farfetched but recent developments in fields like neural networks, computer vision, and natural language processing has paved a way for accurately describing images i.e. representing their visually grounded meaning.

We are leveraging state-of-the-art techniques like Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and appropriate datasets of images and their human perceived description to achieve the same.

We demonstrate that our alignment model produces results in retrieval experiments on datasets such as Flickr.

## 2.3 OPEN PROBLEMS IN THE EXISTING SYSTEM

- The main aim of this project is to get a little bit of knowledge of deep learning techniques. We use two techniques mainly CNN and LSTM for image classification.
- So, to make our image caption generator model, we will be merging these architectures. It is also called a CNN-RNN model.
- CNN is used for extracting features from the image. We will use the pre-trained model Xception.
- LSTM will use the information from CNN to help generate a description of the image.

## 2.4 EXISTING SYSTEM

- The entire flow of the model was explained from data set collection to caption generation.
- Here Support Vector Method Algorithm was used as the encoder and Logistic Regression was used as the decoder for generating the caption.
- As we have seen in the literature survey there are many drawbacks to the existing model.
- Each existing model has its own disadvantage making the model less efficient and less accurate when the results are generated.

### DEEP LEARNING-BASED IMAGE CAPTIONING METHODS:

We draw an overall taxonomy in Figure 1 for deep learning-based image captioning methods. We discuss their similarities and dissimilarities by grouping them into visual space vs. multimodal space, dense captioning vs. captions for the whole scene, Supervised learning vs. Other deep learning, Encoder-Decoder architecture vs. Compositional architecture, and one „Others” group that contains Attention-Based, Semantic Concept-Based, Stylized captions, and Novel Object-Based captioning.

We also create a category named LSTM vs. Others. A brief overview of the deep learning-based image captioning methods is shown in the table. It contains the name of the image captioning methods, the type of deep neural networks used to encode image information, and the language models used in describing the information. In the final column, we give a category label to each captioning technique based on the taxonomy in Figure 2.1.

## 2.5 OTHER DEEP LEARNING-BASED IMAGE CAPTIONING

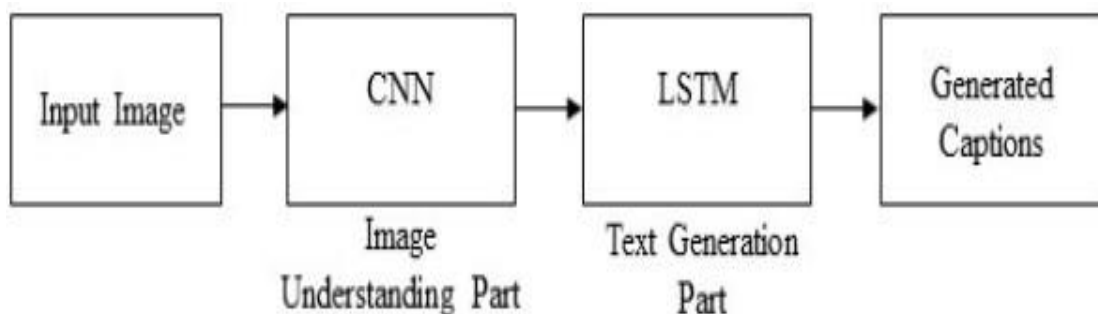
In our day-to-day life, data are increasing with unlabeled data because it is often impractical to accurately annotate data. Therefore, recently, researchers are focusing more on reinforcement learning and unsupervised learning-based techniques for image captioning.

### 2.5.1 DENSE CAPTIONING:

The previous image captioning methods can generate only one caption for the whole image. They use different regions of the image to obtain information of various objects. However, these methods do not generate region-wise captions. Johnson et al. [12] proposed an image captioning 16 methods called Dense Caption. This method localizes all the salient regions of an image and then it generates descriptions for those regions.

A typical method of this category has the following steps:

- (1) Region proposals are generated for the different regions of the given image.
- (2) CNN is used to obtain the region-based image features.
- (3) The outputs of Step 2 are used by a language model to generate captions for every region. A block diagram of a typical dense captioning method is given in Figure 2.1.



***Fig.2.1 A Block Diagram of Simple Encoder-Decoder Architecture-Based Image Captioning.***

### 2.5.2 CAPTIONS FOR THE WHOLE SCENE

Encoder-Decoder architecture, Compositional architecture, attention-based, semantic concept-based, stylized captions, Novel object-based image captioning, and other deep learning networks-based image captioning methods generate single or multiple captions for the whole scene.

## ENCODER-DECODER ARCHITECTURE VS. COMPOSITIONAL ARCHITECTURE:

Some methods use just a simple vanilla encoder and decoder to generate captions. However, other methods use multiple networks for it.

### **Encoder-Decoder Architecture-Based Image Captioning**

The neural network-based image captioning methods work as just simple end to end manner. These methods are very similar to the encoder-decoder framework-based neural machine translation [11]. In this network, global image features are extracted from the hidden activations of CNN and then fed them into an LSTM to generate a sequence of words. A typical method of this category has the following general steps:

- (1) A vanilla CNN is used to obtain the scene type, to detect the objects and their relationships.
- (2) The output of Step 1 is used by a language model to convert them into words, combined phrases that produce image captions.

A simple block diagram of this category is given in Figure 2.2

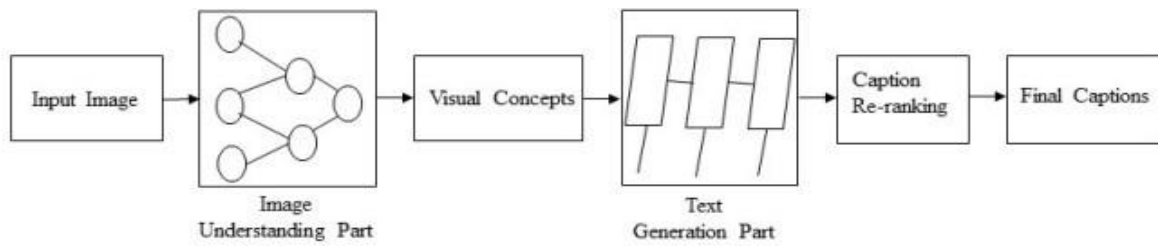
### COMPOSITIONAL ARCHITECTURE-BASED IMAGE CAPTIONING:

Compositional architecture-based methods composed of several independent functional building blocks: First, a CNN is used to extract the semantic concepts from the image. Then a language model is used to generate a set of candidate captions. In generating the final caption, these candidate captions are re-ranked using a deep multimodal similarity model.

A typical method of this category maintains the following steps:

- (1) Image features are obtained using a CNN.
- (2) Visual concepts (e.g., attributes) are obtained from visual features.
- (3) Multiple captions are generated by a language model using the information from Step 1 and Step 2.
- (4) The generated captions are re-ranked using a deep multimodal similarity model to select high-quality image captions.

A common block diagram of compositional network-based image captioning methods is given in Figure 5.



**Fig.2.2. A block diagram of a compositional network-based captioning**

## 2.6 LSTM VS. OTHERS

Image captioning intersects computer vision and natural language processing (NLP) research. NLP tasks, in general, can be formulated as sequence-to-sequence learning. Several neural language models such as neural probabilistic language model, log-bilinear models, skip-gram models, and recurrent neural networks (RNNs) have been proposed for learning sequence-to-sequence tasks. RNNs have widely been used in various sequence learning tasks.

LSTM units use a memory cell that can maintain information in memory for long periods of time. In recent years, LSTM-based models have dominantly been used in sequence-to-sequence learning tasks. Another network, Gated Recurrent Unit (GRU) has a similar structure to LSTM but it does not use separate memory cells and uses fewer gates to control the flow of information. However, LSTMs ignore the underlying hierarchical structure of a sentence.

They also require significant storage due to long-term dependencies through a memory cell. In contrast, CNNs can learn the internal hierarchical structure of the sentences and they are faster in processing than LSTMs.

Therefore, recently, convolutional architectures are used in other sequence to sequence tasks, e.g., conditional image generation and machine translation. Inspired by the above success of CNNs in sequence learning tasks, proposed a CNN language model-based image captioning method. This method uses a language-CNN for statistical language modelling.

However, the method cannot model the dynamic temporal behavior of the language model only using a language-CNN.

It combines a recurrent network with the language CNN to model the temporal

dependencies properly. proposed a convolutional architecture for the task of image captioning.

They use a feed-forward network without any recurrent function. The architecture of the method has four components:

- (i) input embedding layer
- (ii) image embedding layer
- (iii) convolutional module, and
- (iv) output embedding layer.

It also uses an attention mechanism to leverage spatial image features. They evaluate their architecture on the challenging MSCOCO dataset and shows comparable performance to an LSTM based method on standard metrics.

## 2.7 MOSTLY REFERRED RESEARCH PAPERS

[1] Sherstinsky, Alex. "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network." *Physical D: Nonlinear Phenomena* 404 (2020): 132306. Print.

This paper provides a mathematical understanding and differentiation between Recurring Neural Networks (RNN) and Long Short-Term Memory (LSTM). An RNN model can be defined using Delay Differential Equations. An RNN can be transformed into an LSTM. This paper explains RNN and LSTM fundamental concepts.

**Summary:** This article is used to develop an LSTM model for our project which will be used for generating captions as a sequence processor.

[2] Liu, Yu Han. "Feature Extraction and Image Recognition with Convolutional Neural Networks." *Journal of Physics: Conference Series* 1087 (2018): 062032. Print.

In the age of Machine Learning and Artificial Intelligence, Image recognition is one of the most important tools. Its uses ranges from searching of images, to tagging objects in social media or vehicle driving assistant systems. The basic problem is to determine if an image has any object or not, and if it does have then which category does it belongs to i.e., feature extraction. That is where Convolution Neural Networks (CNN) comes into use. It is a type of feed-forward Artificial Neural Network which is widely used for image processing.

**Summary:** This article helps us in understanding and using the Convolution Neural



Networks which we will use for feature extraction from the images.

[3] Aung, San & Pa, Win & nwe, tin. (2020). "Automatic Myanmar Image Captioning using CNN and LSTM-Based Language Model." Proceedings of the 1st Joint SLTU and CCURL Workshop (SLTU-CCURL 2020), pages 139–143. Print.

A natural scene image contains objects, colors, activities, attributes etc. Humans can differentiate between these elements of an image instantaneously. But, for a machine, it is a difficult task. Image captioning requires 2 components: Identifying objects and attributes, and understanding the relationships between those objects. We use a merged model with CNN and LSTM to generate the captions for Burmese language captions.

**Summary:** In this paper, we learn the importance and the need of a merged model to generate image captions for the Burmese language which can be generalized for English as well.

[4] Chollet, Francois. "Xception: Deep Learning with Depth-wise Separable Convolutions." 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017). Print.

An architecture in which Inception modules are replaced with depth-wise separable convolutions is known as an Xception model. A depth-wise separable convolution is an inception model but with a very large number of towers. Xception architecture is proven to outperform Inception(v3) architecture on a small dataset but the performance improvements are very high on large datasets. They both have the same number of parameters. Hence, the performance gain is due to higher efficiency.

**Summary:** In this paper, we learned the importance of the Xception model. How Xception and Inception(v3) models differ from each other.

## **CHAPTER 3**

### **REQUIREMENT ANALYSIS**

#### **3.1 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. The feasibility study investigates the problem, and the information needs of the stakeholders. It seeks to determine the resources required to provide an information systems solution, the cost, and benefits of such a solution, and the feasibility of such a solution.

The goal of the feasibility study is to consider alternative information systems solutions, evaluate their feasibility, and propose the alternative most suitable to the organization. The feasibility of a proposed solution is evaluated in terms of its components.

##### ***3.1.1 ECONOMICAL FEASIBILITY***

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

##### ***3.1.2 TECHNICAL FEASIBILITY***

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

### **3.1.3 SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

## **3.2 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT**

### **HARDWARE REQUIREMENTS**

- Hard Disk -160GB
- Key Board -Standard Windows Keyboard
- Mouse-Two or Three-Button Mouse
- Monitor-SVGA
- Processor-I3/Intel Processor
- System-Windows
- Speed-2.4GHZ
- RAM-8GB

### **SOFTWARE REQUIREMENTS**

- Operating System - Windows 10 or 11
- Coding language – Python
- IDE – PyCharm, Jupyter Notebook
- Dataset – Flickr8k dataset.
- Domain – Deep Learning
- Web Technologies – Flask, CSS, HTML, JavaScript, SQL.

### **PACKAGES:**

- **OS** - used to handle files using system commands.
- **Pickle** - used to store NumPy features extracted
- **NumPy** - used to perform a wide variety of mathematical operations on arrays
- **tqdm** - progress bar decorator for iterators. Includes a default range iterator printing to stderr.
- **VGG16, preprocess input** - imported modules for feature extraction from the image data
- **load\_img, img\_to\_array** - used for loading the image and converting the image to a NumPy array

- **Tokenizer** - used for loading the text and converting them into a token
- **pad\_sequences** - used for equal distribution of words in sentences filling the remaining spaces with zeros
- **plot\_model** - used to visualize the architecture of the model through different images
- **TensorFlow** - The TensorFlow platform helps you implement best practices for data automation, model tracking, performance monitoring, and model retraining.
- **Keras** - Keras is the high-level API of TensorFlow 2: an approachable, highly productive interface for solving machine learning problems, with a focus on modern deep learning.

### 3.3 LANGUAGE SPECIFICATION

**Python** is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This **tutorial** gives enough understanding on **Python programming** language.

- **What Is A Script?**

Up to this point, I have concentrated on the interactive programming capability of Python. This is a very useful capability that allows you to type in a program and to have it executed immediately in an interactive mode

- **Scripts are reusable**

Basically, a script is a text file containing the statements that comprise a Python program. Once you have created the script, you can execute it over and over without having to retype it each time.

- **Scripts are editable**

Perhaps, more importantly, you can make different versions of the script by modifying the statements from one file to the next using a text editor. Then you can execute each of the individual versions. In this way, it is easy to create different programs with a minimum amount of typing.

- **You will need a text editor**

Just about any text editor will suffice for creating Python script files.

You can use *Microsoft Notepad*, *Microsoft WordPad*, *Microsoft Word*, or just about any word processor if you want to.

### 3.4. APPLICATION OF PYTHON

- **Easy-to-learn** – Python has few keywords, a simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy-to-maintained.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode that allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.
- **GUI Programming** – Python supports GUI applications that can be created and ported to many systems calls, libraries and Windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.
- Apart from the above-mentioned features, Python has a big list of good features, a few are listed below
- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.

- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

### **3.5 DIFFERENT LIBRARIES IN PYTHON**

Python has two basic modes: normal and interactive.

The normal mode is the mode where the scripted and finished .py files are run in the Python interpreter.

Interactive mode is a command line shell that gives immediate feedback for each statement while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole

- **Some Python Libraries:**

- Pandas
- NumPy
- Matplotlib
- Seaborn
- OpenCV
- Keras
- TensorFlow
- NLTK
- Scikit-Learn
- SciPy
- BeautifulSoup
- Text Blob
- Pillow
- Request
- SQLAlchemy
- PyTorch

- Selenium

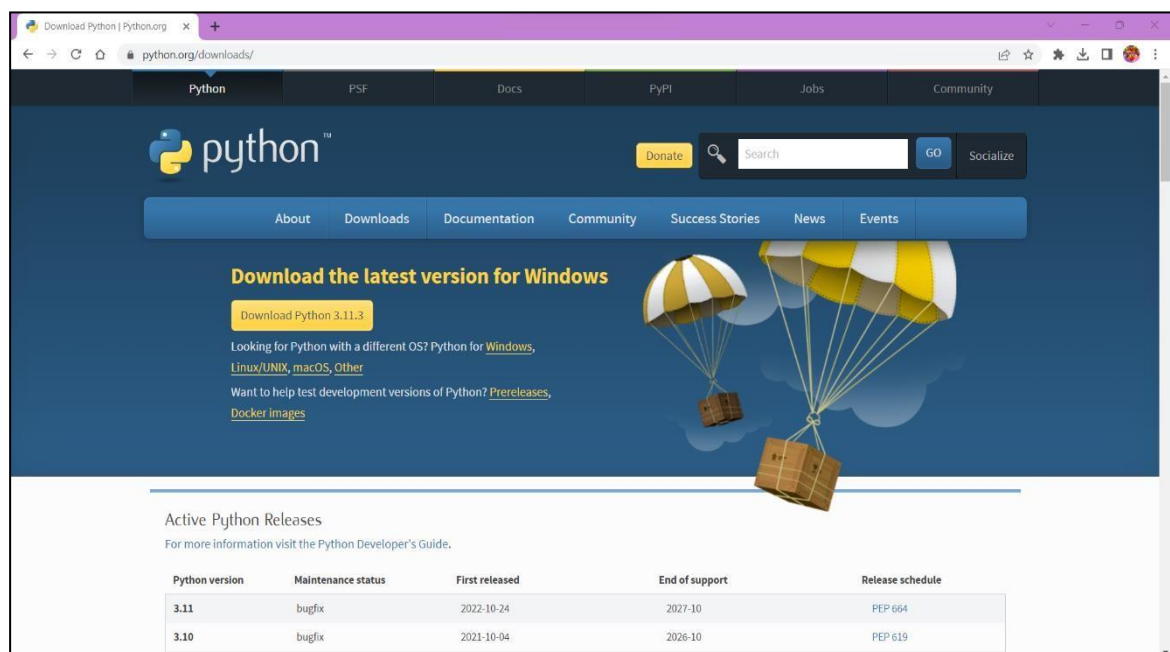
### 3.6 FEATURES OF PYTHON

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to bytecode for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

### 3.7 SOFTWARE INSTALLATION FOR IMAGE CAPTION GENERATION PROJECTS

#### 3.7.1 INSTALLING PYTHON:

1. To download and install Python visit the official website of Python <https://www.python.org/downloads/> and choose your version.



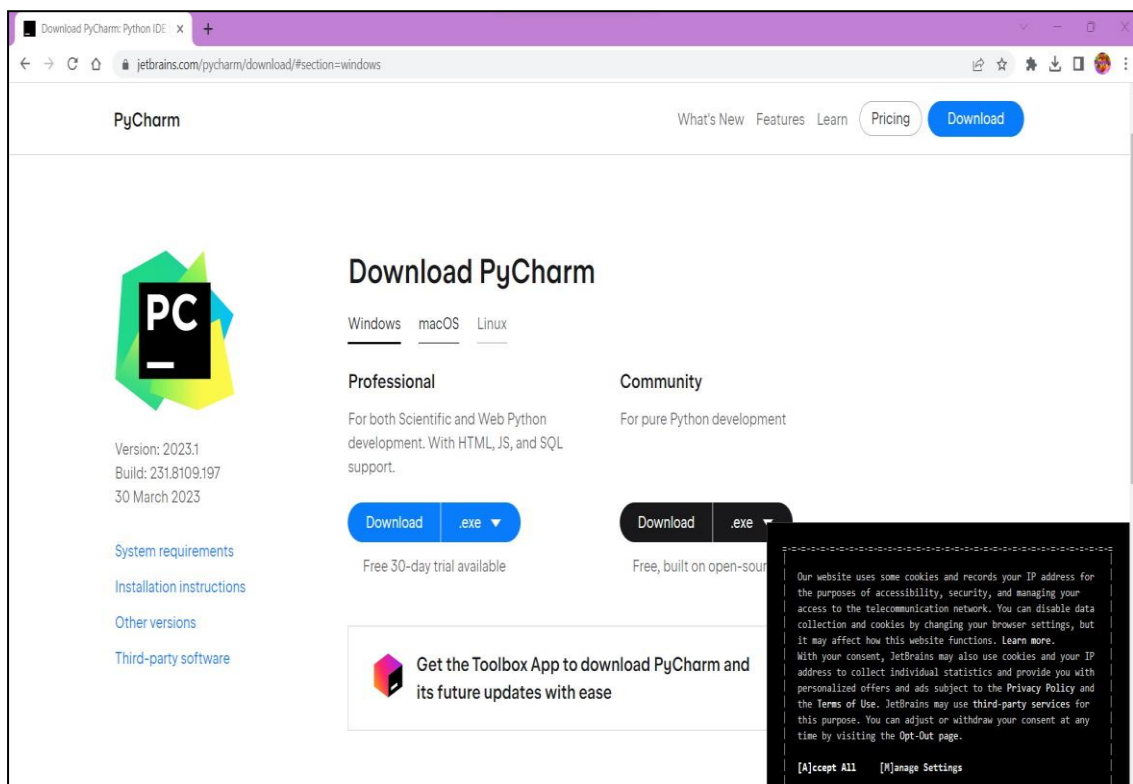
**Fig.3.1. A block diagram of a Python installation**

2. Once the download is complete, run the exe for installing Python. Now click on Install Now.

3. You can see Python installed at this point.
4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

### 3.7.2 INSTALLING PYCHARM:

1. To download PyCharm visit the website <https://www.jetbrains.com/pycharm/download/> and Click the "DOWNLOAD" link under the Community Section.



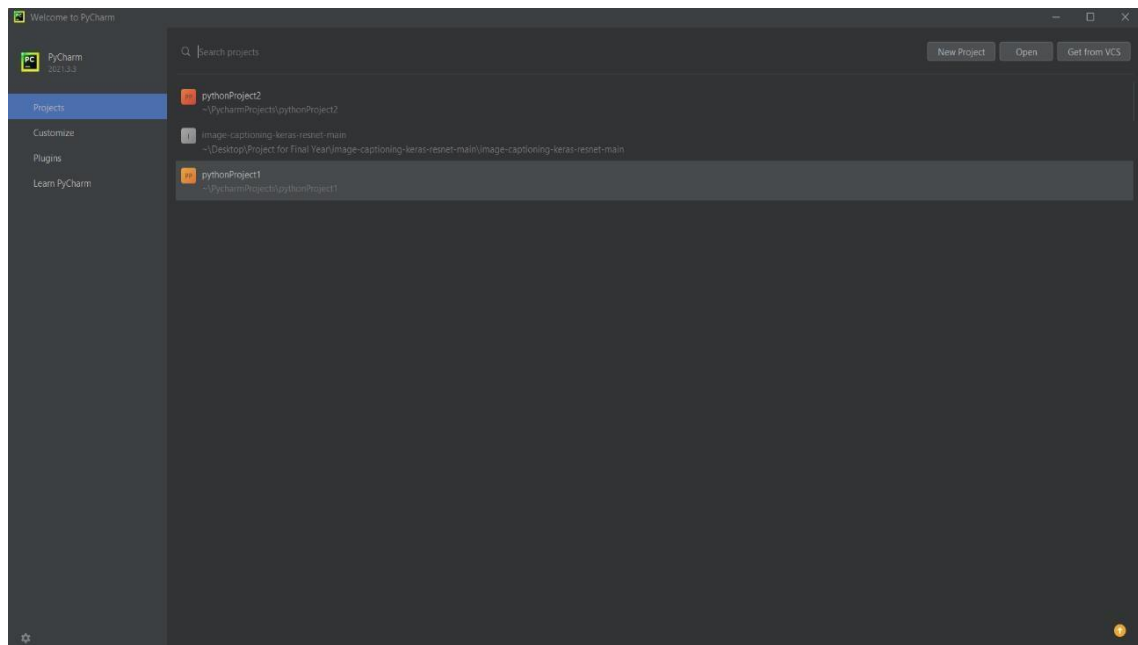
**Fig.3.2. A block diagram of a PyCharm installation**

2. Once the download is complete, run the exe to install PyCharm. The setup wizard should have started. Click "Next".
3. On the next screen, Change the installation path if required. Click "Next".
4. On the next screen, you can create a desktop shortcut if you want and click on "Next".
5. Choose the start menu folder. Keep selecting JetBrains and click on "Install".
6. Wait for the installation to finish.
7. Once installation is finished, you should receive a message screen that



PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.

8. After you click on "Finish," the Following screen will appear.



**Fig.3.3. A block diagram of a create new project on PyCharm**

9. You need to install some packages to execute your project in a proper way.
10. Open the command prompt/ anaconda prompt or terminal as administrator.
11. The prompt will get open, with specified path, type “pip install package name” which you want to install (like NumPy, pandas, seaborn, scikit-learn, matplotlib.pyplot)

Ex: pip install NumPy

```
C:\WINDOWS\system32>pip install numpy==1.18.5
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp36-cp36m-win_amd64.whl (12.7 MB)
    |████████████████████████████████████████| 12.7 MB 939 kB/s
ERROR: tensorboard 2.0.2 has requirement setuptools>=41.0.0, b
Installing collected packages: numpy
Successfully installed numpy-1.18.5
```

### 3.8 SYSTEM USECASES

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form, UML is comprised of two major components:

a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of a software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

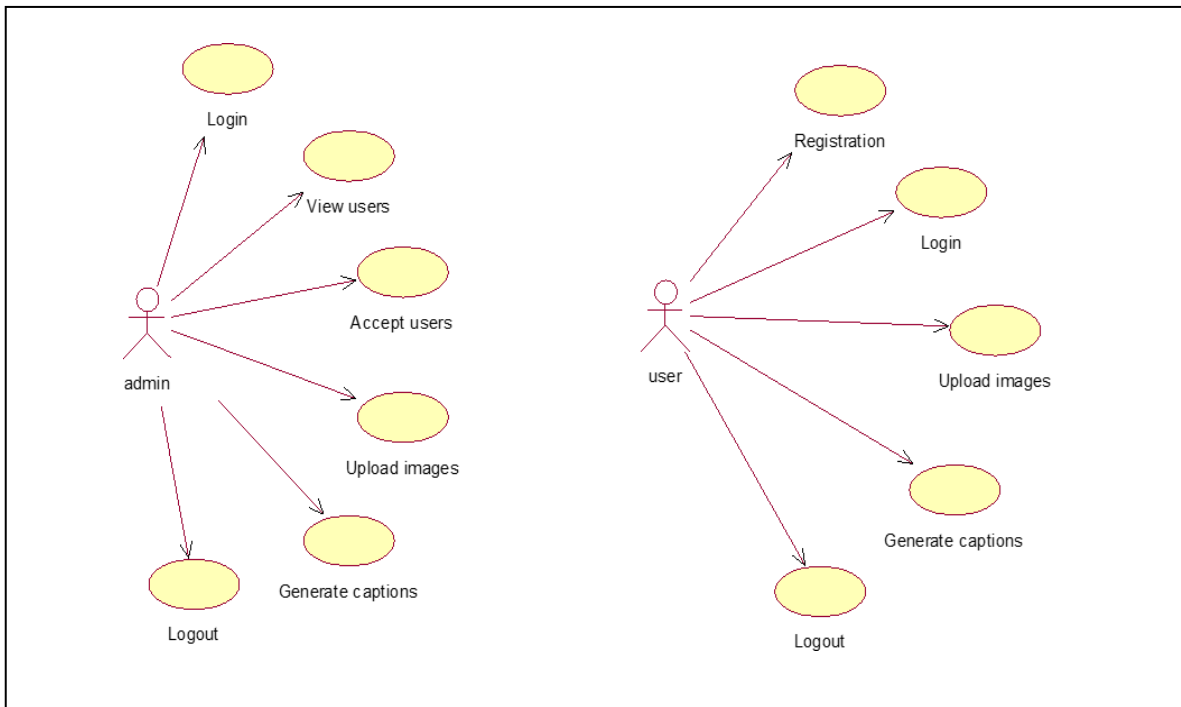
#### **GOALS:**

The Primary goals in the design of the UML are as follows:

- Provide users with a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extensibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development processes.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of the OO tools market.
- Support higher-level development concepts such as collaborations, frameworks, patterns and components. Integrate best practices.
- **USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies

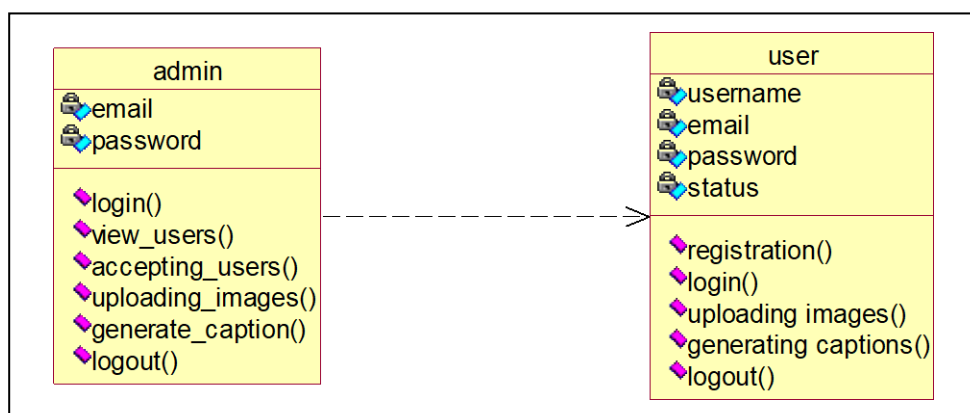
between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



**Fig.3.4. Login Page and Registration use case diagram**

- **CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



**Fig.3.5. A block diagram of a Class Diagram**

### 3.9 IMPORT ALL THE REQUIRED PACKAGE

Install below libraries, to begin with, the project:

```
pip install TensorFlow
```

```
pip install Keras
```

```
pip install pillow
```

```
pip install NumPy
```

```
Pip install tqdm
```

```
Pip install jupyterlab
```

```
import numpy as np
```

```
from PIL import Image
```

```
import os
```

```
import string
```

```
from pickle import dump
```

```
from pickle import load
```

```
from keras.applications.xception import Xception #to get pre-trained model  
Xception
```

```
from keras.applications.xception import preprocess_input
```

```
from keras.preprocessing.image import load_img
```

```
from keras.preprocessing.image import img_to_array
```

```
from keras.preprocessing.text import Tokenizer #for text tokenization
```

```
from keras.preprocessing.sequence import pad_sequences
```

```
from keras.utils import to_categorical
```

```
from keras.layers.merge import add
```

```
from keras.models import Model, load_model
```

```
from keras.layers import Input, Dense#Keras to build our CNN and LSTM
```

```
from keras.layers import LSTM, Embedding, Dropout
```

```
from tqdm import tqdm_notebook as tqdm #to check loop progress
```

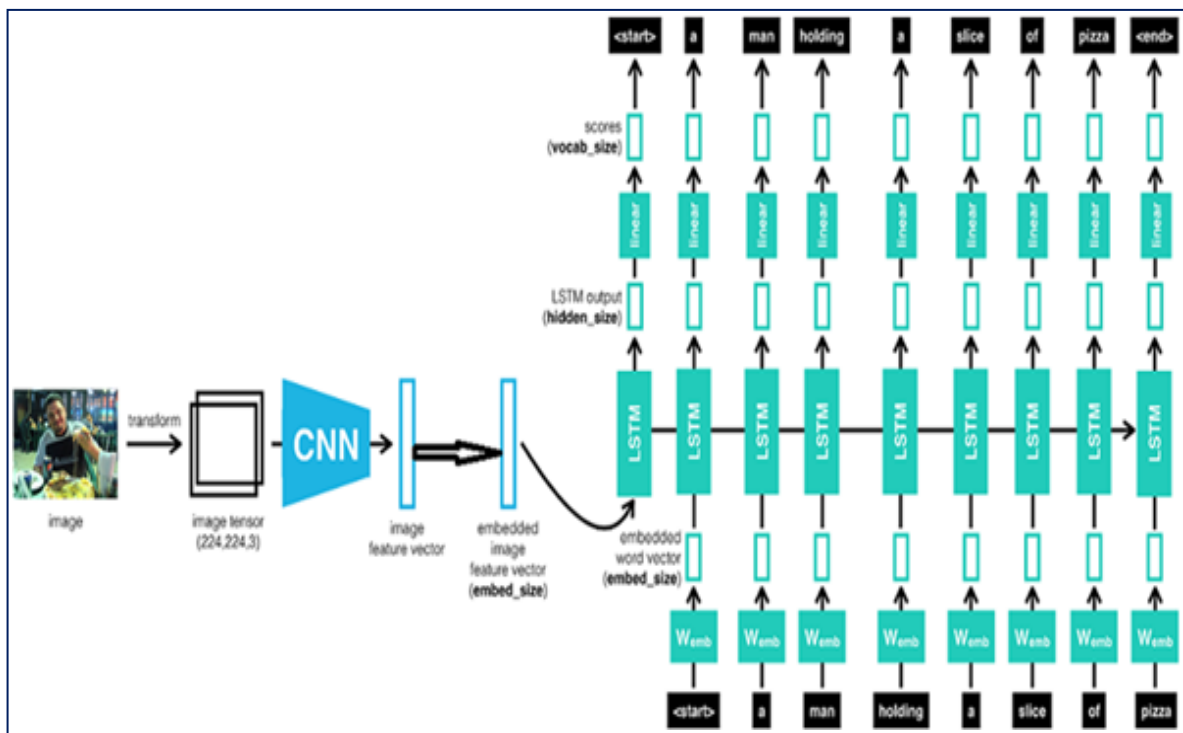
```
tqdm().pandas()
```

## CHAPTER 4

### DESCRIPTION OF PROPOSED SYSTEM

#### 4.1 PROPOSED SYSTEM

- As we have observed that using the traditional CNN-LSTN model there is a vanishing gradient problem that hinders the Recurrent Neural Network to learn and get efficiently trained.
- So, in order to reduce this gradient descent problem, In this paper we are proposing this model so as to increase the efficiency of generating captions for the image and also to increase the accuracy of the captions.
- This Image Captioning can be done by using Deep Learning Models. With the advancement of deep learning and Natural Language Processing now it has become easy to generate captions for the given images.



**Fig.4.1 Real-Time Image Captioning on An Embedded System Using A Light-Weight Deep Learning**

## 4.2 SELECTED METHODOLOGY OR PROCESS MODEL

### 4.2.1 CNN ALGORITHM

**1. Input Layer:** It is the layer where we provide the input for the model. The number of features our input has is equal to the number of neuron in the input layer.

**2. Hidden Layer:** The input features are transferred to the hidden layer(s) where different processes/activities takes place. There can be multiple hidden layers. The layers undergoes mathematical operations like matrix multiplication, convolutions, pooling etc. along with an activation function.

**3. Output Layer:** They layer which is used to generate probability scores using sigmoid or SoftMax functions which is then converted to the output of our model.

A Convolutional Neural Network (CNN) is a Neural Network with at least one convolutional layers. They are used for various purposes including image processing, feature extraction, classification etc. Instead of taking an entire image at once to find features, it is more effective to look at small parts of the image. CNN is commonly used for feature extraction from images. CNNs also perform well at image segmentation, signal processing, speech recognition etc.

A CNN can also be implemented as a U-Net architecture, which are essentially two almost mirrored CNNs resulting in a CNN whose architecture can be presented in a U shape. U-nets are used where the output needs to be of similar size to the input such as segmentation and image improvement.

Each convolutional layer contains a series of filters known as convolutional kernels. The filter is a matrix of integers that are used on a subset of the input pixel values, the same size as the kernel. Each pixel is multiplied by the corresponding value in the kernel, then the result is summed up for a single value for simplicity representing a grid cell, like a pixel, in the output channel/feature map. These are linear transformations; each convolution is a type of affine function.

In computer vision the input is often a 3 channel RGB image. For simplicity, if we take a greyscale image that has one channel (a two-dimensional matrix) and a 3x3 convolutional kernel (a two-dimensional matrix). The kernel strides over the

input matrix of numbers moving horizontally column by column, sliding/scanning over the first rows in the matrix containing the images pixel values. Then the kernel strides down vertically to subsequent rows.

- **Image Input Layer:**

Create an image input layer using `image_input_layer`. An image input layer inputs images to a network and applies data normalization. Specify the image size using the input `Size` argument. The size of an image corresponds to the height, width, and the number of color channels of that image. For example, for a grayscale image, the number of channels is 1, and for a color image it is 3.

- **Convolution Layer:**

Convolutional layers are the major building blocks used in convolutional neural networks.

A convolution is the simple application of a filter to an input that results in an activation. Repeated application of the same filter to an input results in a map of activations called a feature map, indicating the locations and strength of a detected feature in an input, such as an image.

The innovation of convolutional neural networks is the ability to automatically learn a large number of filters in parallel specific to a training dataset under the constraints of a specific predictive modeling problem, such as image classification. The result is highly specific features that can be detected anywhere on input images.

- **Pooling Layer:**

It is common to periodically insert a Pooling layer in-between successive Convolution layer in a CNN architecture. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation. The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2 down samples every depth slice in the input by 2

along both width and height, discarding 75% of the activations. Every MAX operation would in this case be taking a max over 4 numbers (little 2x2 region in some depth slice). The depth dimension remains unchanged.

- **Input Sequence Layer:**

A sequence input layer inputs sequence data to a network. In the general case, input sequences and output sequences have different lengths (e.g. machine translation) and the entire input sequence is required in order to start predicting the target. This requires a more advanced setup, which is what people commonly refer to when mentioning "sequence-to-sequence models" with no further context. Here's how it works:

In the general case, input sequences and output sequences have different lengths (e.g. machine translation) and the entire input sequence is required in order to start predicting the target. This requires a more advanced setup, which is what people commonly refer to when mentioning "sequence-to-sequence models". Here's how it works:

- A RNN layer (or stack thereof) acts as an "encoder": it processes the input sequence and returns its own internal state. Note that we discard the outputs of the encoder RNN, only recovering the state.
- Another RNN layer (or stack thereof) acts as a "decoder": it is trained to predict the next characters of the target sequence, given previous characters of the target sequence.

- **SoftMax and Classification Layers:**

SoftMax converts logits into probabilities by taking the exponents from every output and then norms each of these numbers by the sum of such exponents, such that the entire output vector adds up to one – every probability should be one. Generally, cross-entropy loss is the loss of such a problem in several classes. In the last layer of an image classification network such as CNN (e.g., VGG16) used in ImageNet competitions, SoftMax is also applied.

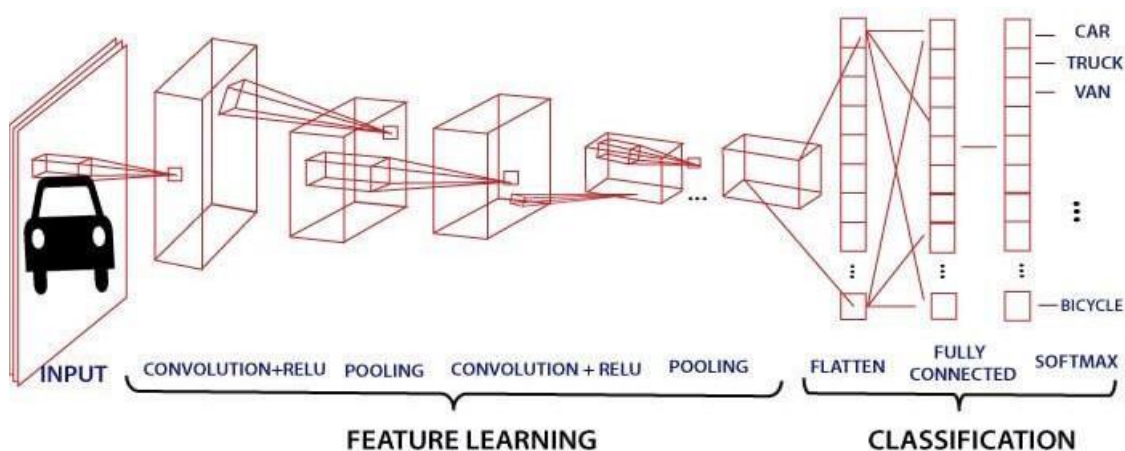
A SoftMax layer applies a SoftMax function to the input. A classification layer



computes the cross-entropy loss for multi-class classification problems with mutually exclusive classes. Create a classification layer using classification Layer. For classification problems, a SoftMax layer and then a classification layer must follow the final fully connected layer. The SoftMax function is also known as the normalized exponential and can be considered the multi-class generalization of the logistic sigmoid function. For typical classification networks, the classification layer must follow the SoftMax layer. In the classification layer, train Network takes the values from the SoftMax function.

CNN takes an image as input, which is classified and process under a certain category such as dog, cat, lion, tiger, etc. The computer sees an image as an array of pixels and depends on the resolution of the image. Based on image resolution, it will see as  $h * w * d$ , where  $h$ = height  $w$ = width and  $d$ = dimension. For example, An RGB image is  $6 * 6 * 3$  array of the matrix, and the grayscale image is  $4 * 4 * 1$  array of the matrix.

In CNN, each input image will pass through a sequence of convolution layers along with pooling, fully connected layers, and filters (Also known as kernels). After that, we will apply the Soft-max function to classify an object with probabilistic values 0 and 1.



**Fig.4.2 Convolution Layer**

Convolution layer is the first layer to extract features from an input image. By learning image features using a small square of input data, the convolutional layer preserves

the relationship between pixels. It is a mathematical operation which takes two inputs such as image matrix and a kernel or filter.

- The dimension of the image matrix is  $h \times w \times d$ .
- The dimension of the filter is  $f_h \times f_w \times d$ .
- The dimension of the output is  $(h-f_h+1) \times (w-f_w+1) \times 1$ .

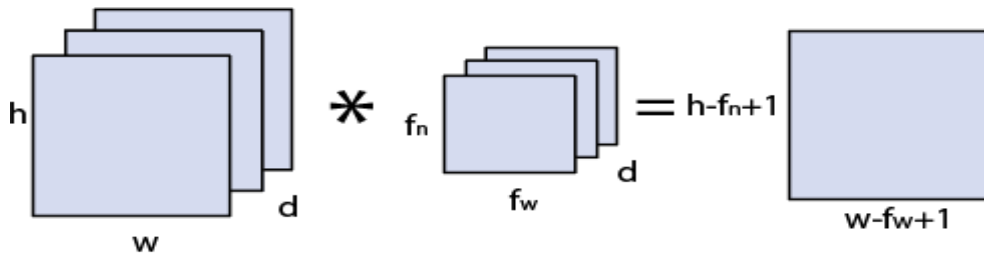


Image matrix multiplies kernel or filter matrix

Let's start with consideration a 5\*5 image whose pixel values are 0, 1, and filter matrix 3\*3 as:

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

5 × 5 – Image Matrix      3 × 3 – Filter Matrix

The convolution of 5\*5 image matrix multiplies with 3\*3 filter matrix is called "**Features Map**" and show as an output.

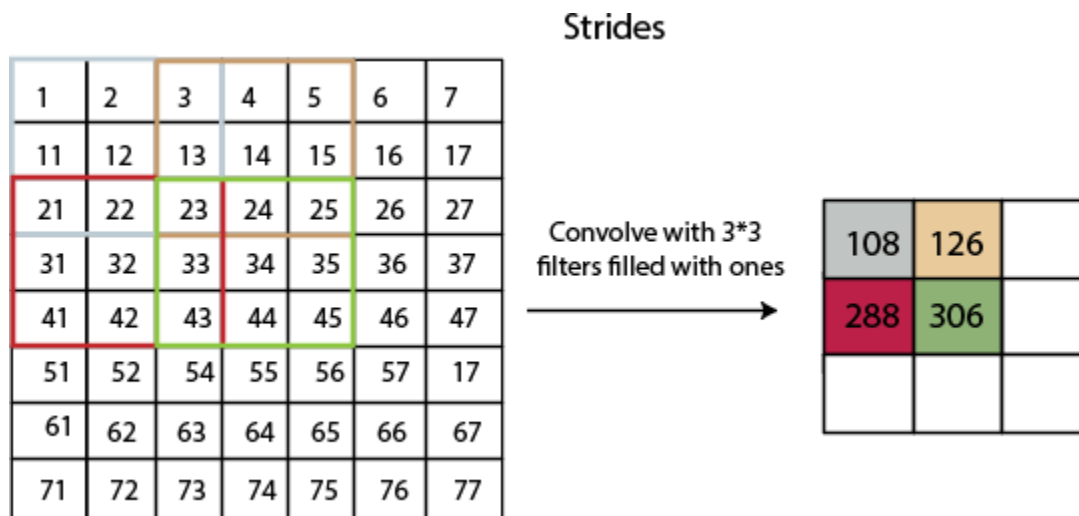
$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 3 & 4 \\ 2 & 4 & 3 \\ 2 & 3 & 4 \end{bmatrix}$$

**Convolved Feature**

Convolution of an image with different filters can perform an operation such as blur, sharpen, and edge detection by applying filters.

- **STRIDES**

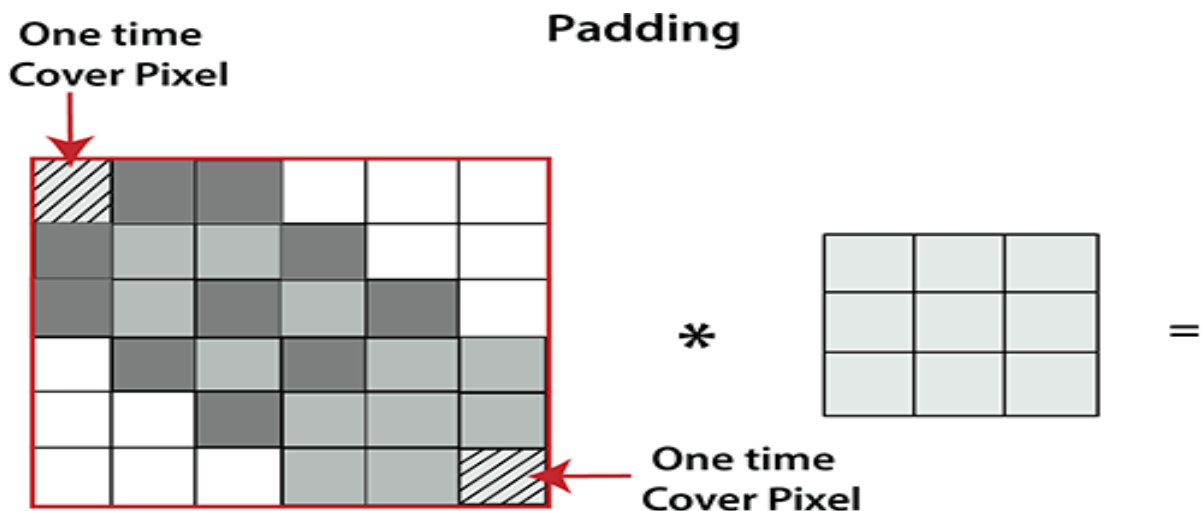
Stride is the number of pixels which are shift over the input matrix. When the stride is equaled to 1, then we move the filters to 1 pixel at a time and similarly, if the stride is equaled to 2, then we move the filters to 2 pixels at a time. The following figure shows that the convolution would work with a stride of 2.



- **PADDING**

Padding plays a crucial role in building the convolutional neural network. If the image will get shrink and if we will take a neural network with 100's of layers on it, it will give us a small image after filtered in the end.

If we take a three-by-three filter on top of a grayscale image and do the convolving then what will happen?



It is clear from the above picture that the pixel in the corner will only get covered one time, but the middle pixel will get covered more than once. It means that we have more information on that middle pixel, so there are two downsides:

- Shrinking outputs
- Losing information on the corner of the image.

To overcome this, we have introduced padding to an image. **"Padding is an additional layer which can add to the border of an image."**

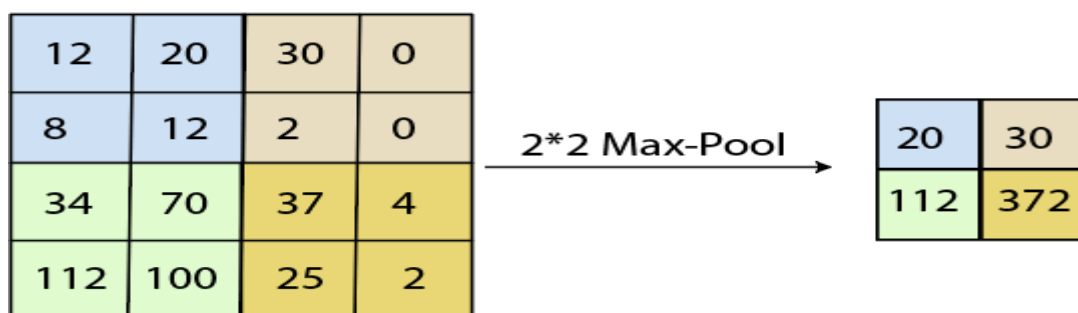
- **POOLING LAYER**

Pooling layer plays an important role in pre-processing of an image. Pooling layer reduces the number of parameters when the images are too large. Pooling is "**downscaling**" of the image obtained from the previous layers. It can be compared to shrinking an image to reduce its pixel density. Spatial pooling is also called downsampling or subsampling, which reduces the dimensionality of each map but retains the important information. There are the following types of spatial pooling:

- **MAX POOLING**

- Max pooling is a **sample-based discretization process**. Its main objective is to downscale an input representation, reducing its dimensionality and allowing for the assumption to be made about features contained in the sub-region binned.
- Max pooling is done by applying a max filter to non-overlapping sub-regions of the initial representation.

### Max Pooling



### AVERAGE POOLING

Down-scaling will perform through average pooling by dividing the input into rectangular pooling regions and computing the average values of each region.

## SYNTAX

```
layer = averagePooling2dLayer(poolSize)
```

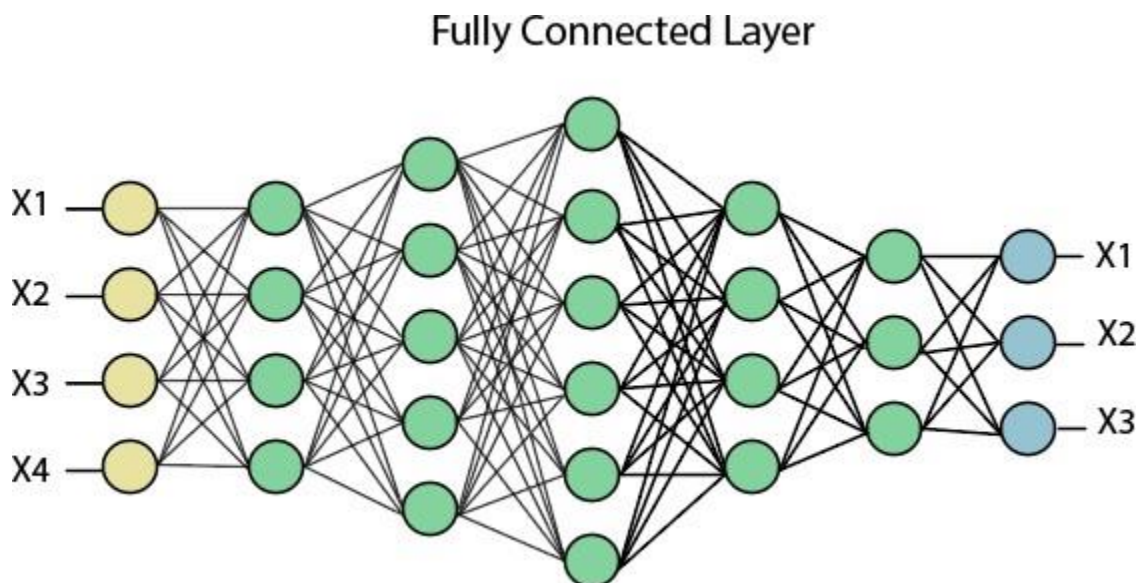
```
layer = averagePooling2dLayer (pool Size, Name, Value)
```

## SUM POOLING

The sub-region for **sum pooling** or **mean pooling** are set exactly the same as for **max pooling** but instead of using the max function we use sum or mean.

- **FULLY CONNECTED LAYER**

The fully connected layer is a layer in which the input from the other layers will be flattened into a vector and sent. It will transform the output into the desired number of classes by the network.



**Fig.4.3 Fully Connected Layer**

In the above diagram, the feature map matrix will be converted into the vector such as **x1, x2, x3... xn** with the help of fully connected layers. We will combine features

to create a model and apply the activation function such as **SoftMax** or **sigmoid** to classify the outputs as a car, dog, truck, etc.

#### **4.2.2 LONG SHORT-TERM MEMORY**

LSTM stands for Long Short-Term Memory they are a type of RNN (Recurrent Neural Network) which is well suited for sequence prediction problems. Based on the previous text, we can predict what the next word will be.

It has proven itself effective from the traditional RNN by overcoming the limitations of RNN which had short term memory. LSTM can carry out relevant information throughout the processing of inputs and with a forget gate, it discards non-relevant information. 21 LSTMs are designed to overcome the vanishing gradient problem and allow them to retain information for longer periods compared to traditional RNNs.

LSTMs can maintain a constant error, which allows them to continue learning over numerous time-steps and backpropagate through time and layers.

**Input gate**—responsible for adding information to the cells

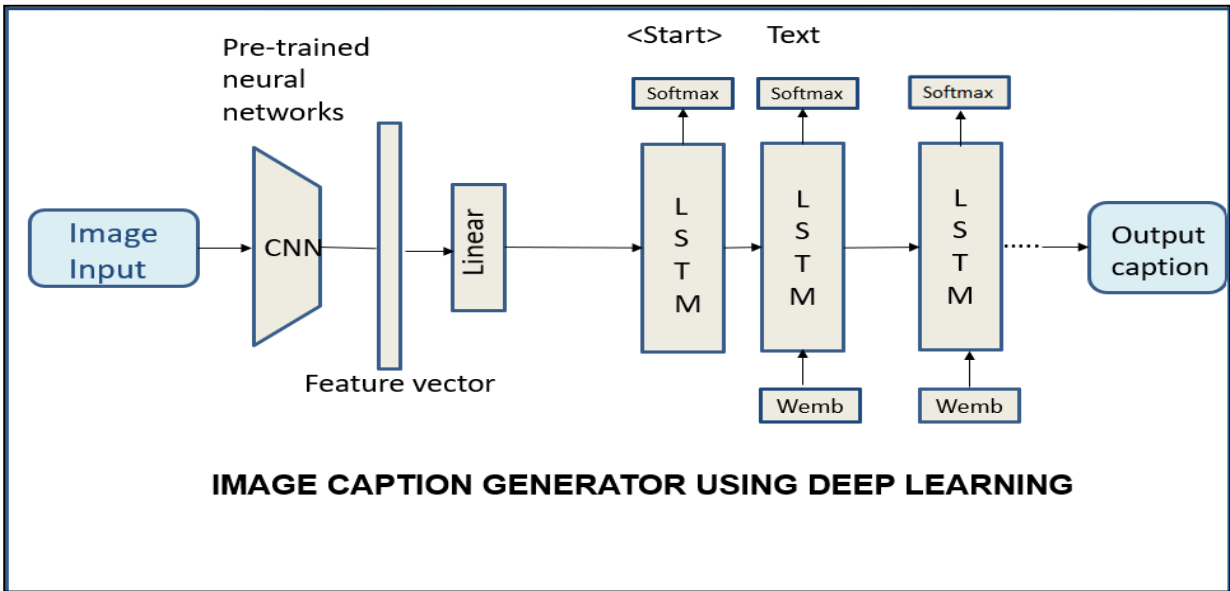
**Output gate**—selects and outputs necessary information

#### **4.3 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM**

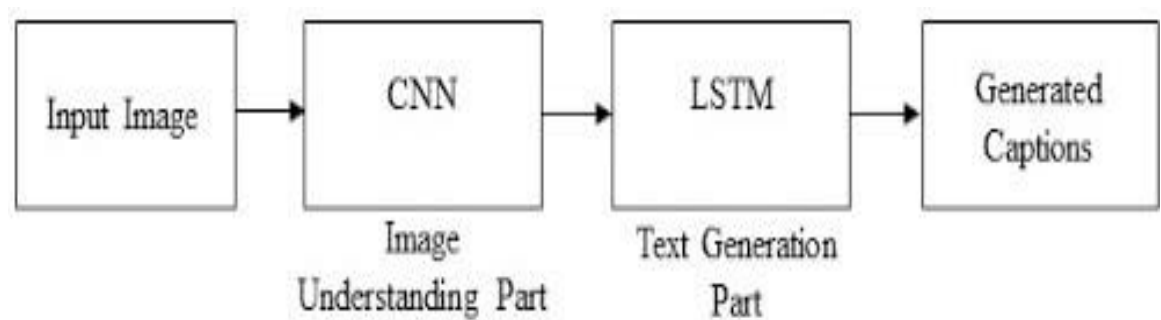
Describing the overall features of the software is concerned with defining the requirements and establishing the high level of the system. During architectural design, the various web pages and their interconnections are identified and designed.

The major software components are identified and decomposed into processing modules and conceptual data structures and the interconnections among the modules are identified. The following modules are identified in the proposed system. The CNN-LSTM architecture involves using CNN layers for feature extraction on input data combined with LSTMs to support sequence prediction.

This model is specifically designed for sequence prediction problems with spatial inputs, like images or videos. They are widely used in Activity Recognition, Image Description, Video Description and many more.



**Fig.4.4 General Architecture Of The CNN - LSTM Model**



**Fig.4.5 Image Caption Generator Using Deep Learning**

#### 4.4 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

##### 4.4.1 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing

can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

#### **4.4.2 OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output.

It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, status or projections of the Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action

#### **4.4.3 PLAN OF PROPOSED SYSTEM**

This Image Captioning can be done by using Deep Learning Models. With the advancement of deep learning and Natural Language Processing now it has become easy to generate captions for the given images.

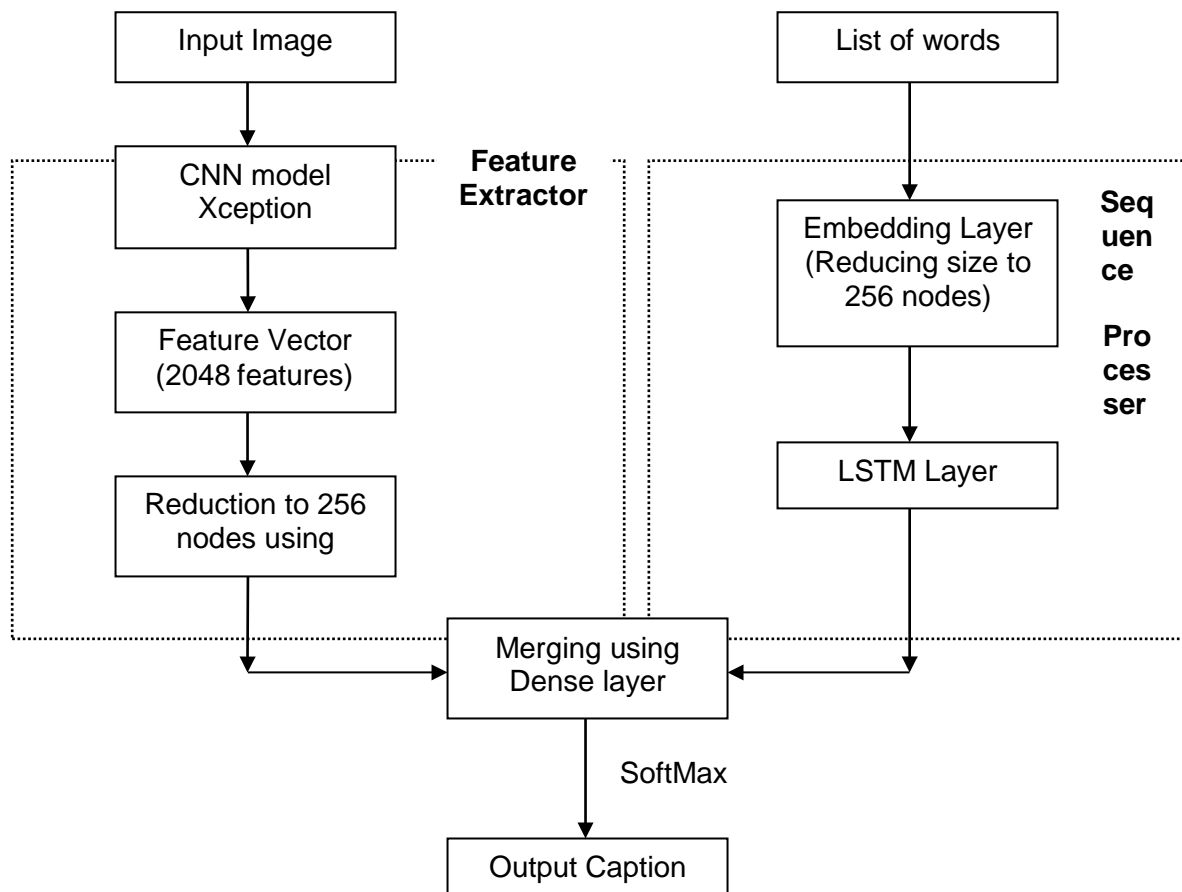
We propose a modified image caption generator which uses Xception model from keras library instead of Inception v3. Since, we are using Flickr8k dataset (which is



a considerably small dataset), the difference between our proposed system and the existing system is minimal. But, if we use large datasets like flickr30k, MSCOCO etc., the increase in computational efficiency will be highly significant. We will generate captions by merging Xception and LSTM model.

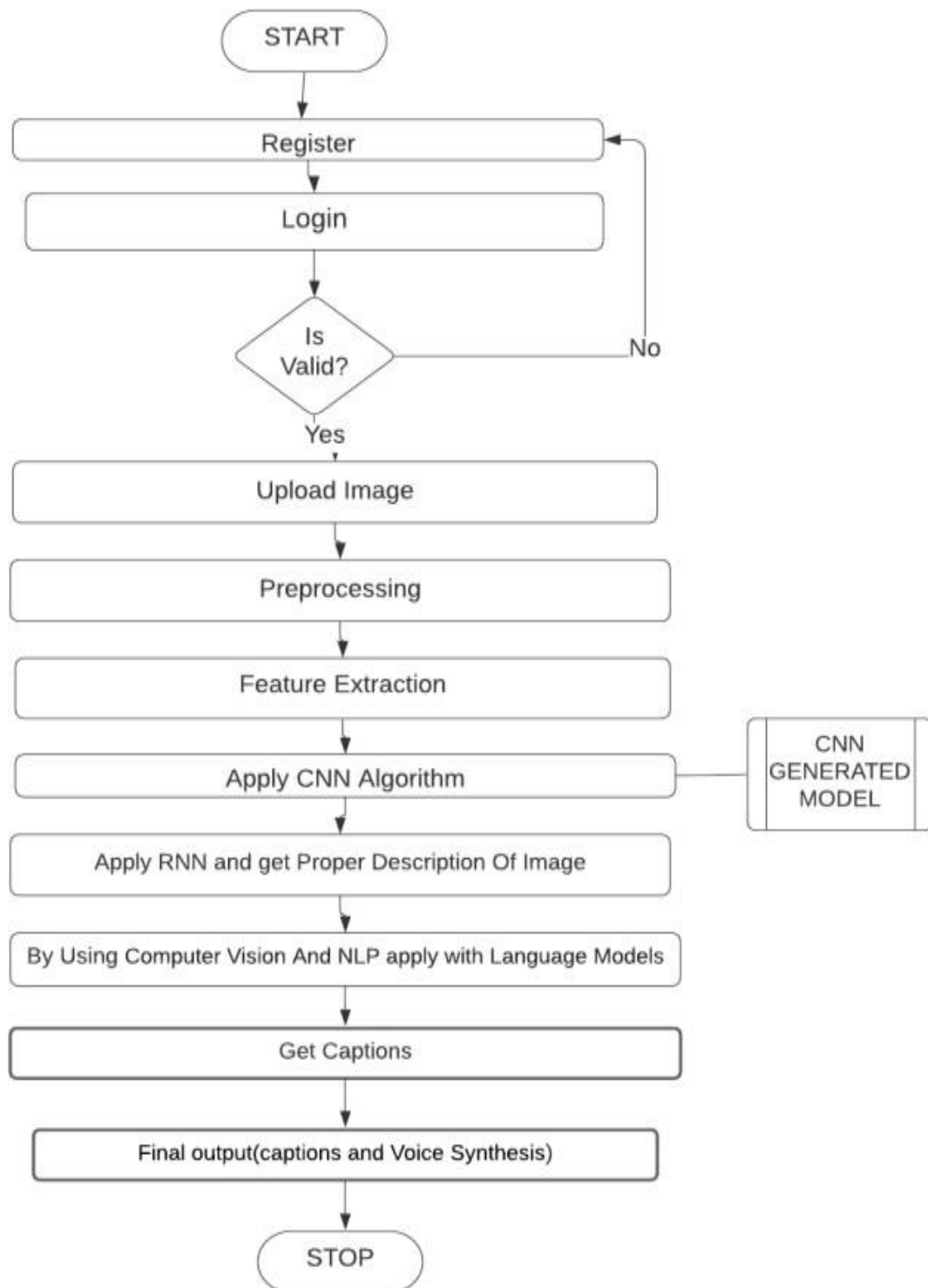
**Advantages:**

- Higher Accuracy especially for large datasets.
- Low computational complexities.



**Fig.4.6 plan of proposed model architecture**

## 4.5 PROJECT MANAGEMENT PLAN



**Fig.4.7 Project Management Plan**

#### **4.5.1 PURPOSE**

The objective of our project is to learn the concepts of a CNN and LSTM model and build a working model of Image caption generator by implementing CNN with LSTM.

##### **Applications:**

- Visually Impaired people.
- Social media companies like Facebook, Twitter etc.
- Search engine for image search like Google, Yahoo, Bing, DuckDuckGo etc.
- Autonomous vehicle developing companies like Tesla, Google, Uber, Baidu etc.

#### **4.5.2 SCOPE**

The scope of this Generating a caption for a given image is a challenging problem in the deep learning domain. In this article, we will use different techniques of computer vision and NLP to recognize the context of an image and describe them in a natural language like English. we will build a working model of the image caption generator by using CNN (Convolutional Neural Networks)

- Image searching using context.
- Visually impaired can listen to the caption of an Image.
- Classifying different photos in our phone gallery automatically for better management.
- Self-driving vehicles like Tesla, Waymo etc. can use image captioning to better understand the images.
- Social media platforms like Facebook uses image captioning to understand the users and their behaviors.

## CHAPTER 5

### IMPLEMENTATION DETAILS

#### 5.1 DEVELOPMENT AND DEPLOYMENT SETUP

##### 5.1.1 *MODULES*

- **Upload:** Upload the dataset of images (either .jpg or .png) to be read using the OpenCV library.
- **View:** Uploaded dataset can be viewed.
- **Preprocessing:** Data Preprocessing is a technique that is used to convert the raw data into a clean data set. Cleaning the data refers to removing the null values, filling the null values with meaningful values, removing duplicate values, removing outliers, and removing unwanted attributes. If the dataset contains any categorical records means to convert those categorical variables to numerical values.
- **Identifying Features:** Identify the Independent variables by Feature Extraction, on which the captions are dependent.
- **Train and Test Split:** Our data is already separated in 3 parts, training data with 6000 images. Testing data with 1000 images and Validation data with 1000 images.

##### 5.1.2 *BUILDING THE MODEL*

- To understand an image and generate captions, we are proposing a Deep learning-based method.
- Deep learning can provide increased accuracy and decrease in computational power.
- We will use Convolution Neural Networks (CNN) and Long Short-Term Memory (LSTM) to do so.
- Convolution Neural Network (CNN) is widely used for automatic feature extraction from images. It uses the mathematical operator convolution.
- LSTM is used to handle sequences of images at a time.
- It is a type of Recurrent Neural Network (RNN) but RNN have the problem of vanishing gradients which can make it harder to train.

- To counter the problem of vanishing gradients, we implement a model using Long Short-Term Memory (LSTM).
- Long Short-Term Memory (LSTM) cannot just process individual images, but can also process sequences of images or videos.
- It is a type of RNN model.
- An RNN is a type of neural network that is repeated again and again and the newer repetitions are only dependent on the previous computation enabling the processing of a sequence of data.
- The Features from CNN and Sequences from LSTM are merged to generate the captions of any image.
- We merge both CNN and LSTM models to produce the captions.
- SoftMax takes in a vector of numbers and converts them to probabilities which are then used for image-generating results.
- SoftMax converts logits into probabilities by taking the exponents from every output and then normalizing each of these numbers by the sum of such exponents, such that the entire output vector adds up to one.
- **Prediction:** An image in the natural scene is given an input and a caption that describes the image in a sentence is generated.

## 5.2 ALGORITHMS

- **Convolutional Neural Network** is one of the main categories to do image classification and image recognition in neural networks. Scene labeling, objects detections, and face recognition, etc., are some of the areas where convolutional neural networks are widely used.
- Convolution Neural Networks is a type of Deep Neural Network. It is typically used for image visualization purposes. They can be defined as a regularized version of Multi-Layer Perceptron where a multi-Layer perceptron has all of its neurons fully connected which makes increases the likelihood of an overfitted model. It uses the mathematical operator convolution instead of the usual matrix multiplication.

- **LSTM Layer:** Recurrent Neural Networks suffer from short-term memory. If a sequence is long enough, they'll have a hard time carrying information from earlier time steps to later ones. So if you are trying to process a paragraph of text to do predictions, RNN's may leave out important information from the beginning.

During back propagation, recurrent neural networks suffer from the vanishing gradient problem. Gradients are values used to update a neural networks weight. The vanishing gradient problem is when the gradient shrinks as it back propagates through time. If a gradient value becomes extremely small, it doesn't contribute too much learning.

### 5.3 SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

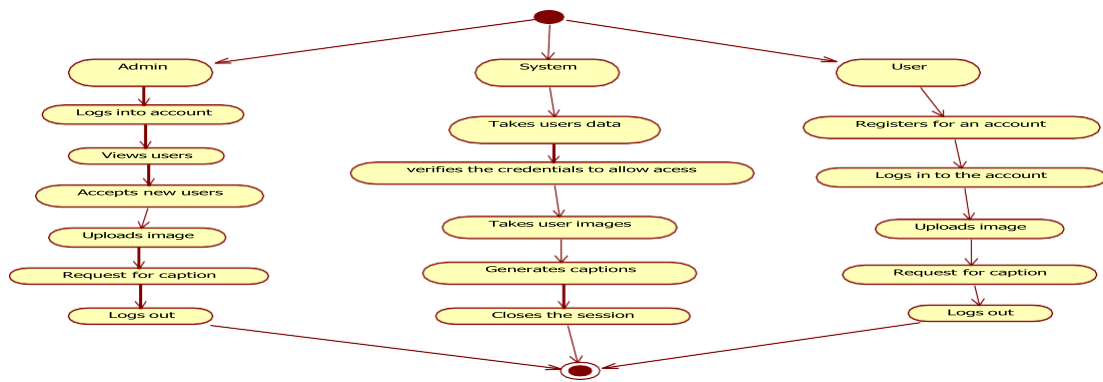
- **Testing code:** As indicated above, code is usually developed in a file using an editor.
- To test the code, import it into a Python session and try to run it.
- Usually there is an error, so you go back to the file, make a correction, and test again.
- This process is repeated until you are satisfied that the code works. The entire process is known as the development cycle.
- There are two types of errors that you will encounter. Syntax errors occur when the form of some command is invalid.
- This happens when you make typing errors such as misspellings, or call something by the wrong name, and for many other reasons. Python will always give an error message for a syntax error.

## 5.4 STEPS FOR EXECUTING THE PROJECTS

- Import the Libraries/packages.
- Load the labels of the training images.
- Load the training images.
- Load the captions of the training images. (We have 5 captions for each image in the Flickr8k dataset)
- Clean all the captions by lower casing them, removing punctuations etc.
- Save all the captions in one image.
- Compute the set of all unique words in the captions.
- We resize the images.
- Images are normalized.
- We extract features of the images using a pre-trained CNN algorithm known as Xception.
- We generate the vocabulary sequences using the LSTM model.
- CNN and LSTM are merged together.
- The model is then trained with the training dataset of images.
- The model is evaluated by using the testing images.
- The user interface is developed using the Flask architecture.
- HTML5 and CSS3 templates are used to do so.
- The web app includes a login and registration system.
- The data of the users is stored in the MySQL database.
- The generated caption can be viewed in the flask app.

## 5.5 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

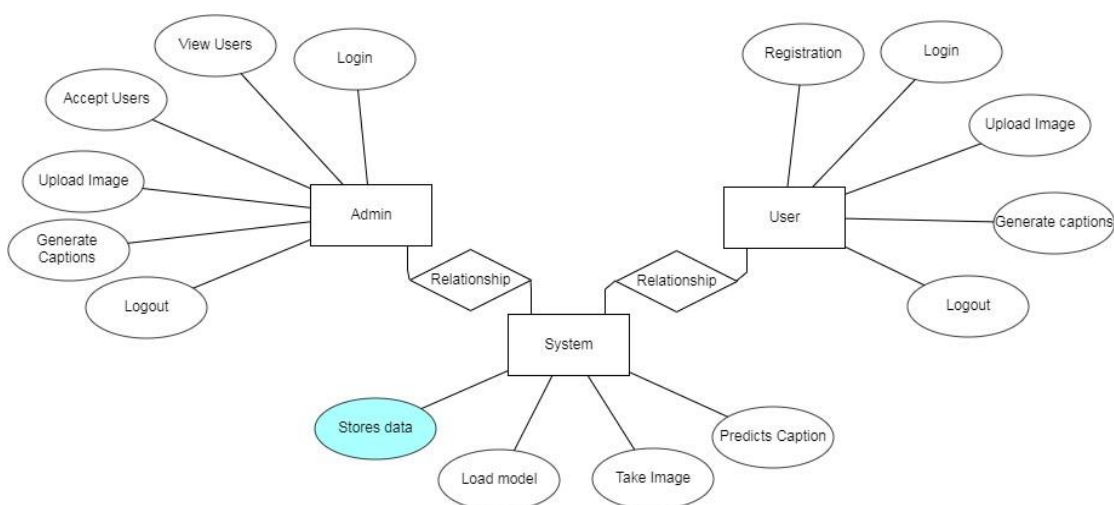


**Fig.5.1 Workflow of Project**

## 5.6 ER DIAGRAM:

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as an Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of the E-R model are entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in the database, so by showing the relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let’s have a look at a simple ER diagram to understand this concept.



**Fig.5.2 Relationship Model Diagram**

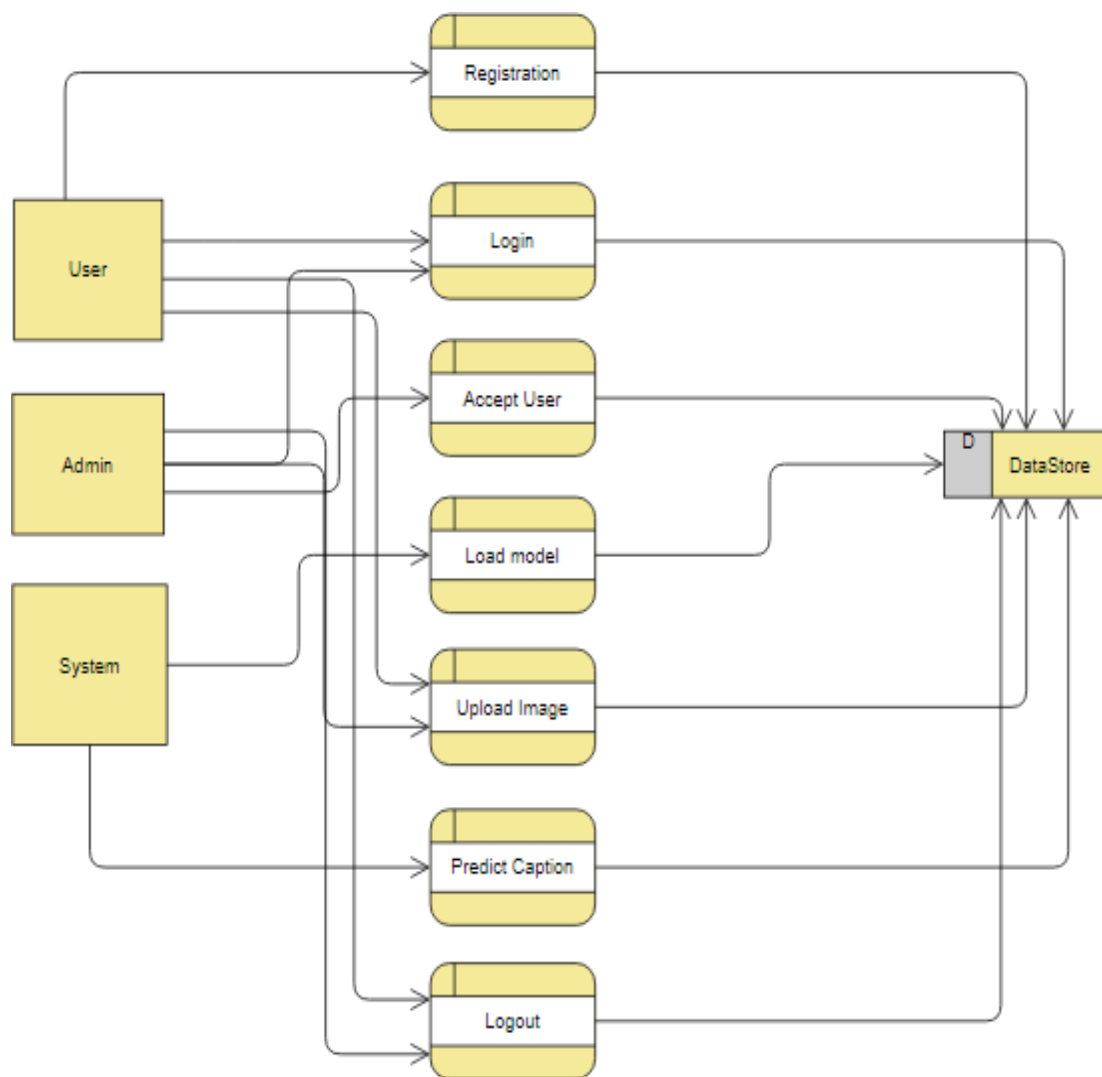


## CHAPTER 6

### RESULTS AND DISCUSSION

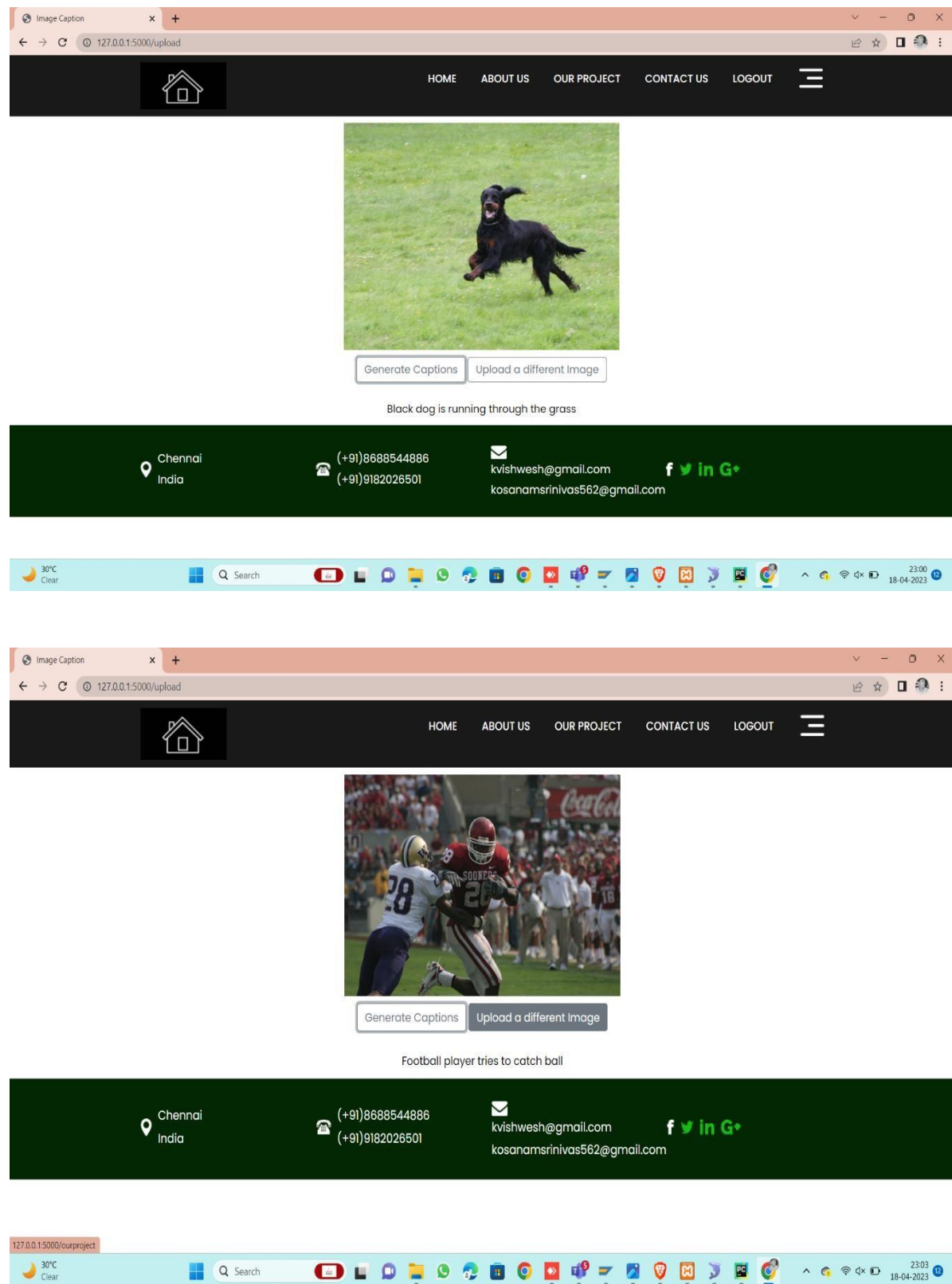
#### 6.1 CAPTION GENERATION FOR IMAGE

The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.



***Fig.6.1 Process of Application***

## 6.2 RESULT OF IMAGE CAPTION GENERATION



**Fig.6.2 Result of Image caption generation**

## **CHAPTER 7**

### **CONCLUSION**

#### **7.1 CONCLUSION**

We have successfully developed a deep-learning model using the Xception architecture to generate automatic captions. So far, most of the image captioning models have used inception (v3). We have fanned our project in a web-based application using the Flask architecture. Some of the automatically generated captions based on our model are as shown below.

Please note that generating captions is subjective and captions for the same image can differ from person to person. This is also the reason why the algorithm which is trained on human-typed captions can generate erratic results sometimes.

#### **7.2 FUTURE SCOPE**

The accuracy of the current project can be further increased by adding a weightage system to the vocabulary by assigning low weight to highly frequently occurring words. It can potentially create a better algorithm to generate captions.

#### **7.3 RESEARCH ISSUES**

There are many challenges and open problems in image captioning which are a part of the problem's nature, such as the parallax error. It can even be difficult for the human eye to detect an object at specific angles that change an object's appearance to the point that it is no longer detectable.

#### **7.4 IMPLEMENTATION ISSUES**

- a. Image Segmentation.
- b. Image Classification.
- c. Multiple Aspect Ratios and Spatial Sizes.
- d. Removing Prints or Security like Encryption.
- e. Image Enhancement.

## REFERENCES:

1. Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.
2. Aung, San & Pa, Win & nwe, tin. (2020). "Automatic Myanmar Image Captioning using CNN and LSTM-Based Language Model." *Proceedings of the 1st Joint SLTU and CCURL Workshop (SLTU-CCURL 2020)*, pages 139–143. Print.
3. Ali Furkan Biten, Lluís Gomez, Marc'al Rusinol, and Dimosthenis Karatzas. 2019. Good news, everyone! context driven entity-aware captioning for news images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12466–12475.
4. J. Song, Y. Guo, L. Gao, X. Li, A. Hanjalic, and H. Shen, "From deterministic to generative: multi-modal stochastic RNNS for video captioning," *IEEE Transaction on Neural Networks and Learning System*, vol. 30, no. 10, pp. 3047–3058, 2018.
5. J Vaishnavi; V Narmatha. "Video Captioning based on Image Captioning as Subsidiary Content" *Image caption generation and video captioning. 2022 IEEE*, 2022.
6. J. Aneja, A. Deshpande, and S. Alexander, "Convolutional image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 2018.
7. L.Gao, K. Fan, J. Song, X. Liu, X. Xu, and H. Shen, "Deliberate attention networks for image captioning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 8320–8327, Honolulu, HI, USA, January-February 2019.
8. Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53.
9. Prachi Waghmare, Jayshree katti, et al. "Show and tell: A neural image caption generator." *Computer Vision and Pattern Recognition (CVPR), 2022 IEEE*

- Conference on. IEEE, 2022.
10. Pimpri Chinchwad, Ralf, and N-H. Nagel. "Knowledge representation for the generation of quantified natural language descriptions of vehicle traffic in image sequences." *Image Processing, 2020. Proceedings., International Conference on*. Vol. 2. IEEE, 2022.
  11. P. Anderson, X. He, C. Buehler et al., "Bottom-up and top-down attention for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 2018.
  12. P. Anderson, X. He, C. Buehler et al., "Bottom-up and top-down attention for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 2018.
  13. Sherstinsky, Alex. "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network." *Physica D: Nonlinear Phenomena* 404 (2020): 132306. Print.
  14. Taku Kudo and John Richardson. 2020. Sentence Piece: A simple and language-independent subword tokenizer and detokenizer for neural text processing. In *2018 Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations*, pages 66–71.
  15. Rita Ramos; Bruno Martins. "Using Neural Encoder-Decoder Models With Continuous Outputs for Remote Sensing Image Captioning" *Euro-pean conference on computer vision*. Springer, Berlin, Heidelberg, 2021.
  16. Xiangyu Duan, Hongfei Yu, Mingming Yin, Min Zhang, Weihua Luo, and Yue Zhang. 2019. Contrastive attention mechanism for abstractive sentence summarization. In *2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP/IJCNLP)*, pages 3044–3053
  17. Yang, Yinzhou, et al. "Corpus-guided sentence generation of natural images." *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2019

# APPENDIX

## A. SOURCE CODE

```
main2.py - C:\Image caption Project\Project\main2.py (3.10.8)
File Edit Format Run Options Window Help

import pandas as pd
from flask import Flask, render_template, request, url_for, redirect, flash, send_from_directory, session
from testing_caption_genetator import *
from forms import RegistrationForm, LoginForm
import pymysql
import pymysql.cursors
#from app import app
import os

APP_ROOT=os.path.dirname(os.path.abspath(__file__))
app=Flask(__name__)
#app.secret_key="from infinity to beyond"
app.config['UPLOAD_FOLDER']=os.path.join(APP_ROOT, 'static/image/')
app.config['SECRET_KEY']='b0b4fbefdc48be27a6123605f02b6b86'
#picFolder = os.path.join('static','image')
#app.config['UPLOAD_FOLDER']=picFolder

@app.route("/")
@app.route("/home")
def home():
    return render_template('index.html')

@app.route("/aboutus")
def aboutus():
    return render_template('aboutus.html')

@app.route("/ourproject")
def ourproject():
    return render_template('ourproject.html')

@app.route("/contact")
def contact():
    return render_template('contact.html')

@app.route("/login", methods=[ 'GET', 'POST'])
def login():
    form = LoginForm()

    #Database connection
    db=pymysql.connect(host="localhost", port=3306, user="root", password="", db="image_caption", cursorclass=pymysql.cursors.DictCursor)
    #Table as a data frame
    register=pd.read_sql_query('select * from {}'.format('register'), db)

    if form.validate_on_submit():
        all_emails=register['email']
        if form.email.data in list(all_emails):
            row=all_emails[all_emails==form.email.data]
            index=row.index[0] ## Id is (1 + index)
            if register['status'][index]=='accepted':
                if form.password.data == register['password'][index]:
                    session['logged_in']=True
                    session['admin']=False
                    flash(f'Welcome {register['username'][index]} ! You have been logged in.', 'success')
                    return redirect(url_for('ourproject'))
                else:
                    flash("You password was incorrect. Please try again.", 'warning')
                    return redirect(url_for('login'))
            else:
                flash("Your account has not yet been verified by the admin. Please try after some time.", 'info')
                return redirect(url_for('home'))
        elif form.email.data == 'admin@test.com' and form.password.data == "password":
            .
            ..
            ..
```

```

        flash("You password was incorrect. Please try again.", 'warning')
        return redirect(url_for('login'))
    else:
        flash("Your account has not yet been verified by the admin. Please try after some time.", 'info')
        return redirect(url_for('home'))
    elif form.email.data == 'admin@test.com' and form.password.data == "password":
        session['loggedin'] = True
        session['admin'] = True
        flash("You have been logged in as the Administrator!", 'success')
        return redirect(url_for('ourproject'))
    else:
        flash(f"No account with the email id {form.email.data} exists. Please register now.", 'info')
        return redirect(url_for('register'))
    return render_template('login.html', form=form)

@app.route('/logout')
def logout():
    session.pop('loggedin', None)
    session.pop('admin', None)
    return redirect(url_for('home'))

@app.route('/users')
def users():
    #Database connection
    db=pymysql.connect(host="localhost", port=3306, user="root", password="root", db="image_caption", cursorclass=pymysql.cursors.DictCursor)

    #if request.method=="POST":
    #    user_status=request.form['verify']
    #    cur=db.cursor()
    #    cur.execute(f"UPDATE register SET status={user_status} WHERE id=1")
    #    db.commit()
    # else:
    #Database connection
    db=pymysql.connect(host="localhost", port=3306, user="root", password="", db="image_caption", cursorclass=pymysql.cursors.DictCursor)
    #Table as a data frame
    register=pd.read_sql_query('select * from {}'.format('register'), db)
    #Remove the password column
    register.drop(['password'], axis=1, inplace=True)

    return render_template('users.html', column_names=register.columns.values, row_data=list(register.values.tolist()), link_column="status", zip=zip)

@app.route("/users2/<int:id>/<string:status>", methods=['GET','POST'])
def users2(id, status):
    user_id=int(id)
    user_status=status
    db=pymysql.connect(host="localhost", port=3306, user="root", password="", db="image_caption", cursorclass=pymysql.cursors.DictCursor)
    # Update the record
    try:
        with db.cursor() as cur:
            sql = "update register set status = 'accepted' where id = %s "
            cur.execute(sql, user_id)
            db.commit()
    finally:
        db.close()

    return redirect(url_for('users'))

@app.route("/register", methods=['GET','POST'])
def register():
    form = RegistrationForm()
    #Database connection
    db=pymysql.connect(host="localhost", port=3306, user="root", password="", db="image_caption", cursorclass=pymysql.cursors.DictCursor)

```



```
@app.route('/register', methods=[ 'GET', 'POST' ])
def register():
    form = RegistrationForm()
    #Database connection
    db=pymysql.connect(host="localhost", port=3306, user="root", password="", db="image_caption", cursorclass=pymysql.cursors.DictCursor)
    #Table as a data frame
    register=pd.read_sql_query('select * from {}'.format('register'), db)

    if form.validate_on_submit():
        email=form.email.data
        all_emails=register['email']
        if email in list(all_emails):
            flash('Account already exists with this Email Id! Please Log In.', 'warning')
            db.close()
            return redirect(url_for('login'))
        else:
            #Insert new record
            try:
                with db.cursor() as cur:
                    sql = "INSERT INTO `register` (`username`,`email`,`password`) VALUES (%s, %s, %s)"
                    cur.execute(sql, ((form.username.data), (form.email.data), (form.password.data)))
                db.commit()
            finally:
                db.close()
            flash(f'Account Created for {form.username.data} Sucessfully! Please wait for the admin to verify your account.', 'success')

    return redirect(url_for('login'))
    return render_template('register.html', form=form)

@app.route("/upload", methods=[ "POST" ])
def upload():
    target=os.path.join(APP_ROOT, 'static/image/')

    if not os.path.isdir(target):
        os.mkdir(target)
    file=request.files["myimage"]
    filename=file.filename
    if filename=="":
        flash('No File Selected', 'danger')
        return redirect(url_for('ourproject'))

    destination="/" + join([target, filename])
    #Extension check
    ext = os.path.splitext(destination)[1]
    if (ext==".jpg" or (ext==".png"):
        pass
    else:
        flash("Invalid Extensions! Please select a .jpg or a .png file only.", category="danger")
        return redirect(url_for('ourproject'))

    if not os.path.isfile(destination):
        file.save(destination)

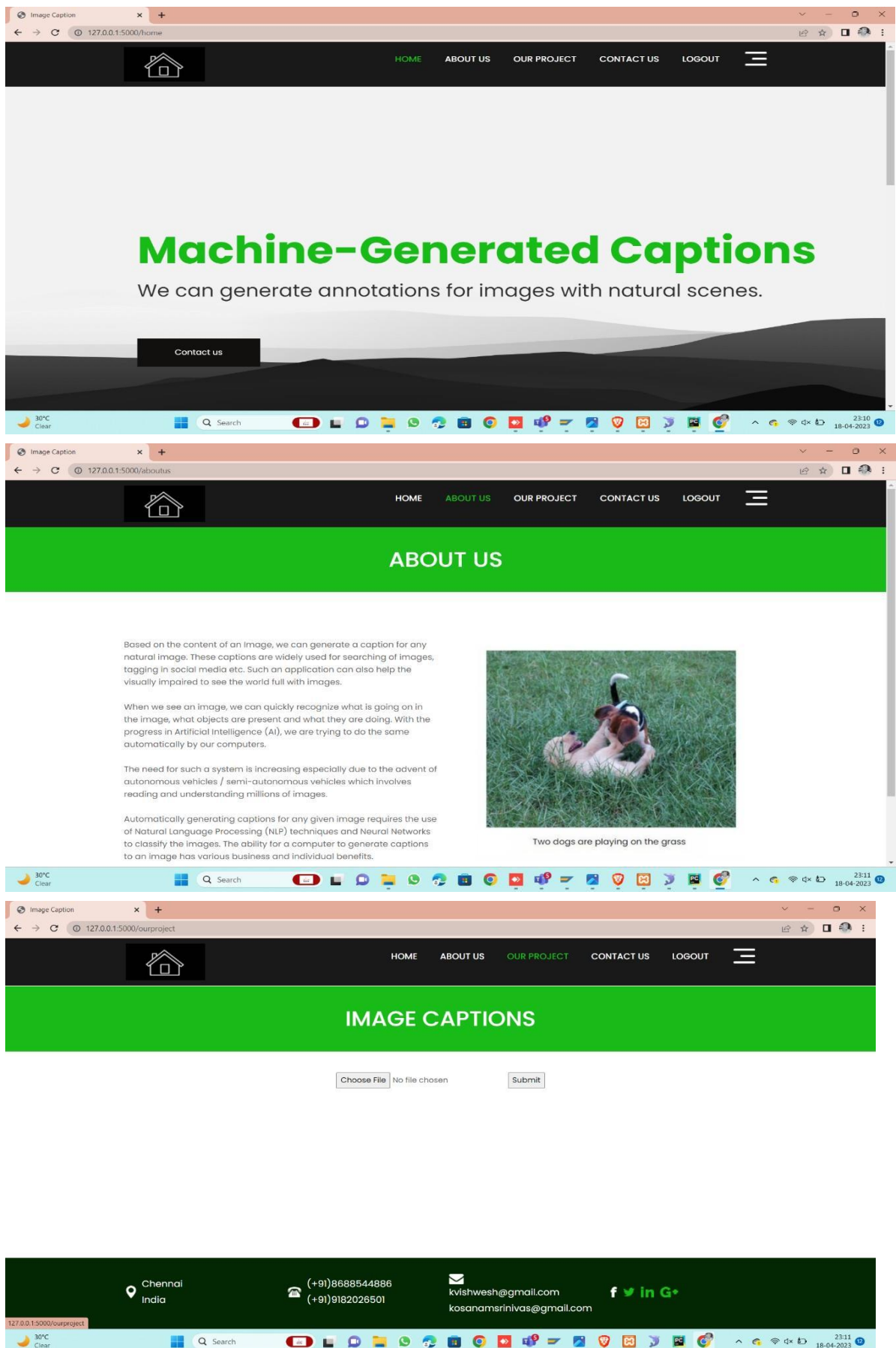
    result=get_caption(destination)
    return render_template("upload.html", img_name=filename, cap=result)

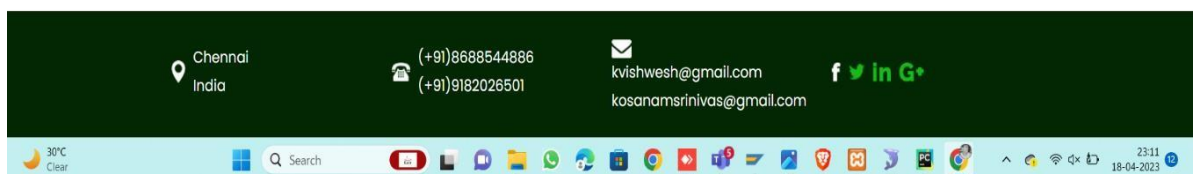
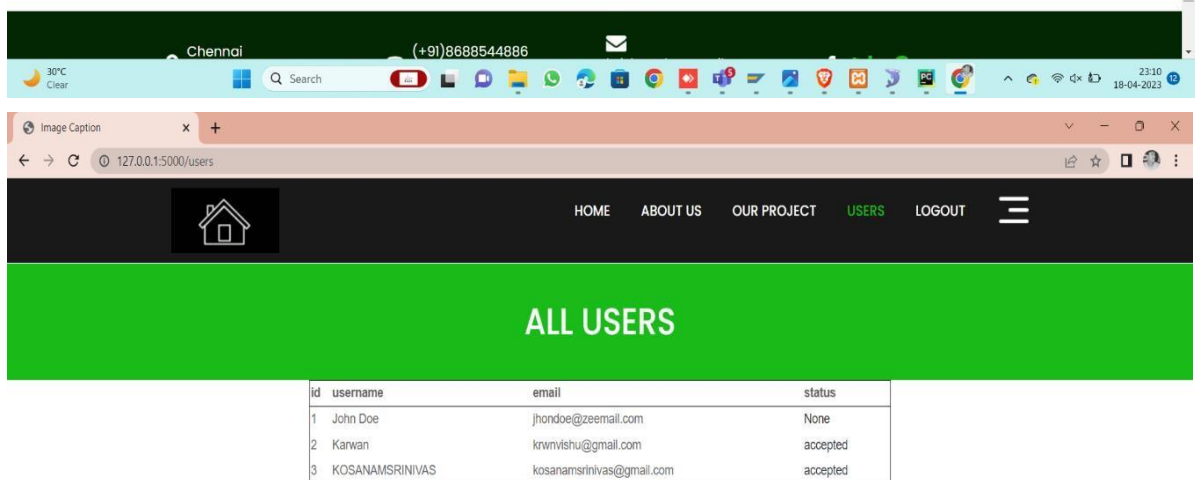
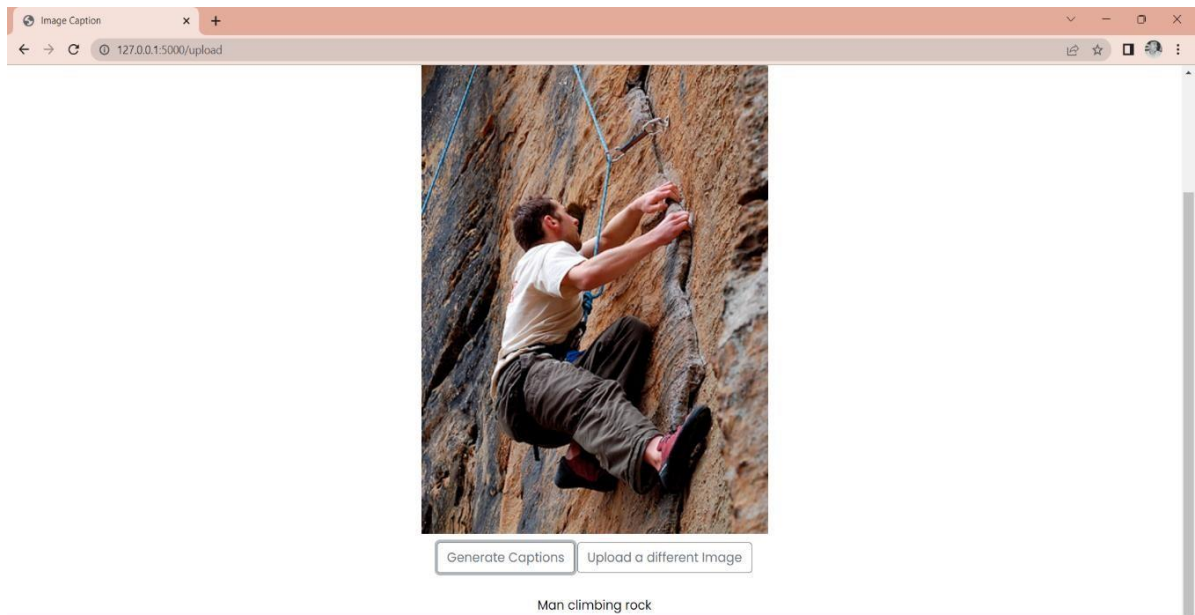
@app.route('/upload/<filename>')
def send_image(filename):
    return send_from_directory(app.config['UPLOAD_FOLDER'], filename)

if __name__ == "__main__":
    app.run(debug=True)
```



## B. SCREENSHOTS





## C. RESEARCH PAPER

# MACHINE-GENERATED CAPTIONS FOR IMAGES USING DEEP LEARNING

Ms.C.A. Daphine Desona Clemency,  
M.E.  
Computer Science and  
Engineering  
Sathyabama Institute of  
Science and  
Technology Chennai, India  
[daphine.cse@sathyabama.ac.in](mailto:daphine.cse@sathyabama.ac.in)

Karwan Vishweshwar  
Computer Science and  
Engineering  
Sathyabama Institute of  
Science and  
Technology  
Chennai, India  
[krwnvishu@gmail.com](mailto:krwnvishu@gmail.com)

Kosanam Srinivas  
Computer Science and  
Engineering  
Sathyabama Institute of  
Science and  
Technology  
Chennai, India  
[kosanamsrinivas562@gmail.com](mailto:kosanamsrinivas562@gmail.com)

**Abstract**—The primary objective of the picture caption generator is to automatically produce a suitable text or caption in English. The system's primary goal is to successfully provide appropriate captions for the provided picture. This study presents an image caption generator that, given an input picture, would identify its contents using beam search and greedy search to produce an English phrase. A pretrained deep learning CNN architecture exception model is used to learn image features, while a LSTM model is used to learn textual features, then integrates the results of both to produce a caption. To produce words, phrases, or captions for the provided photos, we use the LSTM model. Using the Convolutional Neural Network with Long Short-Term Memory, this model was created to create a caption generator for images. Features are extracted from the picture using a pre-trained version of VGG16. To create descriptive text for the pictures, LSTM acts as a decoder. This model has been taught to produce descriptive captions or words based on an input picture. The effectiveness of the model is measured by means of blue scores given to the system. The Keras library, NumPy, and Jupyter notebooks are discussed as tools for developing this project. We also talk about the picture categorization task, how CNNs are employed, and the Flickr dataset.

**Keywords**—: *Deep Learning, LSTM, Caption, Description, Memory, Neural Network, VGG16, Image, CNN*

## I. INTRODUCTION

The field of servo descriptions for images via transfer learning is rapidly growing since it combines the best of artificial intelligence and NLP. The goal of this study is to develop an algorithm to analyze an image's data and provide a human-readable description of it. The initial stage in most efforts is to collect a large dataset consisting of images and their explanations, since it will be used to train the deep learning

algorithm. A popular form of the model combines a run trained to generate language processing tags with a CNN was built to decipher the meaning of a picture. In order to generate captions for images, a CNN must first decide what elements should be retrieved from them. Many other topologies, including converter, multiplexer, even robot architectures, may be used to train this model. By focusing the model's attention on certain parts of the image, transformer-based systems like BERT and GPT-2 may increase caption quality. After the model has been trained, it may be used to generate new captions for unseen images. The degree to which the generated captions match the reference caption may be measured using tools like BLEU, METEOR, ROUGE, and CID. One important challenge is the lack of a large collection of images with text annotations. An additional challenge is developing tags that aren't just comprehensible but also semantically correct and worded in a natural manner. This new technology may be put to use in a variety of contexts, including but not limited to sight searches and retrieval, robotic photo annotation, and devices for the visually impaired. The advancement of computational modeling and natural language recognition has led to an increase in the frequency and quality of machine-generated captions. There are a few approaches that may be taken to automatically generate captions for images using deep learning. The model may also generate captions using a "template-based" approach, in which examples are used as a basis for new captions. This strategy is effective for creating captions for a specific media, such as video games or stock images, but it has its limits. Another approach is the "free-form" technique, in which the model generates captions without relying on a certain format. With this approach, the model can generate

more informative descriptions, although it could be harder to train & evaluate. The advantages of both the framework and unlimited approaches are combined in the "hybrid" method. Using this approach might improve the quality of the captions generated by providing grammatically correct and semantically precise captions

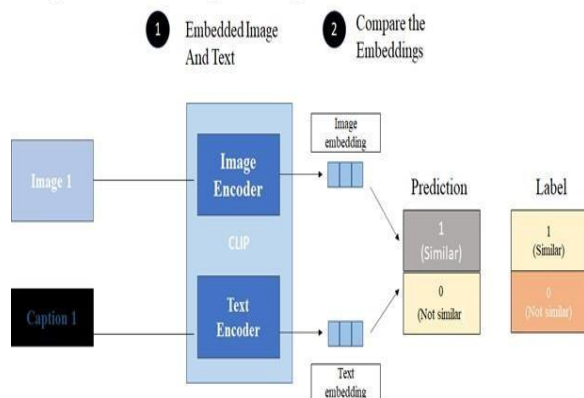


Fig 3. Extract Features From Images And Text

It's also worth noting that various pre-trained models for writing captions for photographs using deep learning are available, including COCO-Caption from Microsoft, Show and Tell from Google, and Dense Captioning from Facebook. Although these methods are already trained, they may be tweaked for optimal performance on specific data. Generally speaking, machine-generated photo captions using deep learning techniques are a rapidly growing field with enormous potential. Future enhancements are expected as a result of research and development efforts in DL and NLP techniques. People communicate with one another via the use of language, whether written or spoken. They usually talk about what they notice using this language. Images and signs may help someone with vision impairments express themselves and learn new information. Automatically creating descriptive sentences from a photograph may help and have a substantial impact on the capacity of visually impaired folks to comprehend the explanation of pictures on the web, despite being a sophisticated and demanding operation [1]. Whenever someone gives a very vivid description of anything, they are said to have painted a "picture" in the listener's mind. The process of creating mental pictures may be quite helpful in developing sentences. Humans can properly identify visuals after just a short exposure. Analysis of existing

natural visual representations may help achieve complex human recognition goals. Image classification & object detection and recognition are far easier problems to solve than automated captioning and description. When characterizing an image, it is important to take into account not only the objects, their characteristics, and the activities they do, but also the connections between them [20]. To this end, most of the early work in facial information has concentrated on assigning labels to images based on specified categories, which has led to tremendous progress. An effective and simple metaphor for assuming is provided by closed visual concept vocabulary. In comparison to the vast mental potential of humans, these concepts seem pitifully little. However, a model of language use in speech perception is necessary, and real dialects like English ought to have been used to convey the aforementioned semantic information. The majority of previous work on automatic description generation from images has argued for a blend of several methods for dealing with the aforementioned problem. As shown in Fig.1, we will instead focus on creating an unified theory that, when fed a picture, can be developed to spit out a set of descriptive terms. The emphasis in natural language processing is often shifted from the importance of images to the way they are represented [18], and this is the case with parsing. One of the main purposes of summarizing is to choose or create a summary for a piece of writing. For the text recognition challenge [21], the objective is to provide a statement that covers a wide range of the visual content.

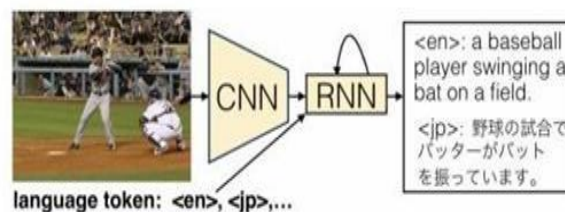
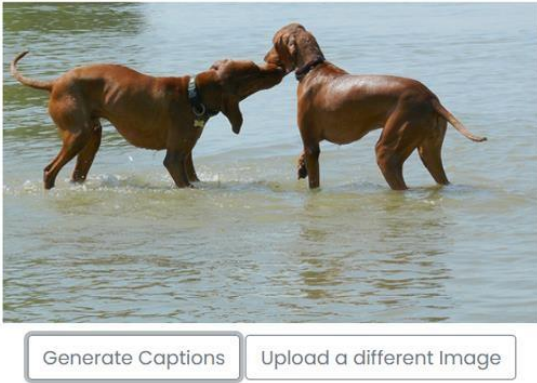


Fig. 1. Model based on Neural Networks

In this study, we offer a paradigm for doing just that; coming up with fresh descriptions of pictures. We've done this using the Flickr 8k dataset, which has 8,000 photos and five descriptions each. Figure 2 depicts the organizational scheme of the dataset, with one picture and its five corresponding natural language descriptions. The approach involves the use of both



CNNs & RNNs. Images are classified using a CNN that has been pre-trained. An image encoder, this network is used to convert input images into a desired format. Recurrent Neural Networks take their input from the last hidden layer (RNN). To put it simply, this network is a translator that can produce new phrases. It appears that the produced sentence sometimes loses its focus or predicts a different phrase than the one that best fits the original visual content. This statement has a tenuous connection to the input picture since it was constructed using a description that occurs often in the dataset.



Two dogs are playing in the water

Fig 2. Caption for Image

## II. LITERATURE REVIEW

Creating natural language descriptions from visual input is a well-studied subject in computer vision [1] [2]. There are essentially three schools of thought in the research around the topic of automatic picture captioning. For starters, there are template-based approaches, which may be found in references [4] - [7]. Object, action, scene, and attribute detection are prioritized in this method. Methods for generating captions that rely on a transfer are the focus of the second group [8]. It is used to retrieve images. Images that are visually comparable are retrieved using this method, and their captions are then applied to the query picture. The majority of studies [10] have shown that using neural networks for translation and artificial neural models and caption creation are both effective. Rather than just translating a text from one language to another, the purpose here is to provide an explanation for the visual. There has been an increase in system complexity as a result of this. They use a formal language to communicate and are composed of

visual radical classifiers (e.g., "and-"). Further transformation may include the usage of a graph or logic system, or a rule-based system. An image's description may be generated using a holistic persistent network model, as proposed by researchers like Mao [11] & Karpathy [12]. Using the NIC model, Vinyals, Oriol. In the NIC paradigm, CNN serves as the encoding method. To classify images, we use an RNN decoder to process output from a pre-trained convolutional neural network (CNN). This RNN decoder has the potential to even generate sentences. For this purpose, we have used LSTM [1], a high-end RNN version. Xu [13] has suggested synthesizing photoreceptor focus into the LSTM process to help keep attention on different things while the algorithm produces appropriate phrases. Generating natural-sounding captions for images involves a complex conventional neural model. All save the most attempting to cut systems employ an encoding-decoding framework [13] that combines caption creation with attention. In this research, we focused on the third category of captioning methods. To do this, a neural system is being developed to provide the visual representations in plain language. The use of CNN for picture encoding is commonplace. After being pre-trained for the task of classifying images, the RNN decoding accepts the data of the last tier as input and outputs the phrase.

## III. OUR APPROACH

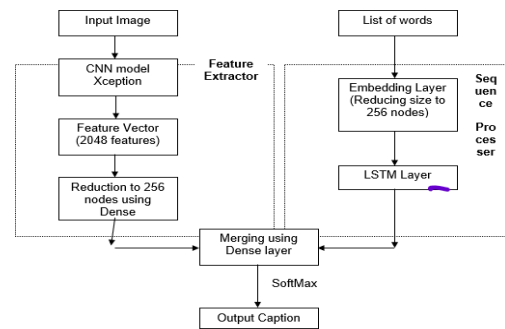


Fig 4. Image Caption Generation System

The purpose of this research was to create an autonomous method for writing picture captions using a neural network trained on probabilistic concepts. By using a powerful statistical model, it may be possible to increase the likelihood of an appropriate translation

occurring during reasoning and learning. A. CNN Today, we use CNN models for anything from speech recognition to facial recognition. The core of a CNN is a series of interconnected convolutional neural networks. A multi-layer neural program's completely connected layers emerge after the inversion procedure [14]. To make advantage of the 2-d nature of the input image, the CNN was designed. To do this, we use a large number of locally - relevant linkages and related data generated using a diversity of pool techniques that are robust to interpretation. CNN's main advantages lie in the fact that it is easy to train and requires fewer settings than other systems with the same number of state variables. For this research [15], we be using a Deep CNN (hence referred to as VGG) net for widespread image recognition. It's available in 16 - & 19- layer variations. Class error values for 19 and 16 layers are really quite similar for both the cross-valid across-validation test set, sitting at 7.4 percent & 7.32 percent corresponding. By feeding an image into the model, it may provide details about it that can be used in the caption generation process. B. LSTM When it comes to NLP & computer vision, recurrent neural networks (RNNs) like the LSTM are often used for tasks like picture captioning. With their ability to "remember" previous data, LSTMs can better grasp the context of new input, making them ideal for sequential data applications like picture captioning. Specifically for the task of visual caption, a Long ShortTerm Memory based model has been trained using a dataset consisting of pictures and their accompanying captions. Using the picture characteristics retrieved and the learnt correlations between image attributes and terms from the trained captions, the model can then produce captions for fresh photos. In order to capture the transient dynamics of a set of objects, researchers have turned to recurrent neural networks [17]. Weights and gradients in a regular RNN tend to vanish and explode, making it hard to understand long- term dynamics [9]. In an LSTM, the "brain" is the storage unit. Existing values are stored for a very long period in the future. The gates' function is to control how often the cell's state is updated. Variations in the amount of connections among memory blocks & gateways stand in for independent variables. Figure 3 depicts the LSTM architecture, upon which our model relies. This design has no peephholes. It can be shown that there are

connections between LSTM's memory cells and their gates in the ways described below:

$$i_l = \sigma(W_{ix}x_l + W_{im}m_{l-1}) \quad (1)$$

$$f_l = \sigma(W_{fx}x_l + W_{fm}m_{l-1}) \quad (2)$$

$$o_l = \sigma(W_{ox}x_l + W_{om}m_{l-1}) \quad (3)$$

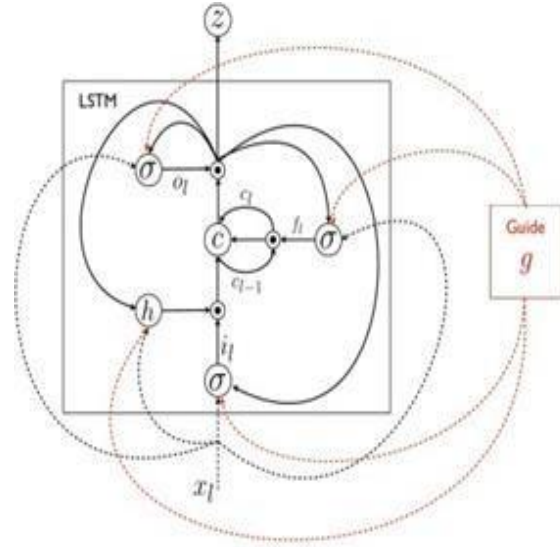


Fig. 3. Connection diagram of LSTM [9]

$$c_l = f_l \odot c_{l-1} + i_l \odot h(W_{cx}x_l + W_{cm}m_{l-1}) \quad (4)$$

$$m_l = o_l \odot c_l \quad (5)$$

$$L(I, S) = - \sum_{t=1}^N \log(p_t(S_t)) \quad (6)$$

#### IV. LSTM-BASED SENTENCE PRODUCTION

The neural network employs an encoder-decoded strategy, used in IOT networks & computational linguistics [1,11,12,13,16] to generate new phrases. Specifically, a set of simple English sentences is encoded as a value is called in this modeling. Next, a decoder uses the translated matrices to generate a new given text in the target language. The learning process's end result is a version that sounds and appears as if it was originally written in the target culture. To maximize the number of captions for a given image, if  $x_i$  is the file's identification &  $s_l$  is the

caption function, this method of creating captions is used. In a well - constructed phrase of length  $L_i$ , the group of phrases  $L_i$  represents the hyperparameters. Moving on, we ignore the superscript  $i$  when it is not relevant or has been taken out of context. The Bayesian cyclical concept may be used to dissect phrases into their individual words, since each is composed of a string.

$$\arg\theta \sum_i \log(p(S_1 : L_i | x^i, \theta))$$

$$\log(p(s_{1:L} | x, \theta)) = \log(p(s_1 | x, \theta)) + \sum_{l=2}^L \log(p(s_l | x, s_{1:l-1}, \theta))$$

where  $S_1:L$  is the part of the constructed sentence up toward the  $l$ th word. To maximize the target in Eq. 7, we specify the document  $\log(p(S_1:L_i | x_i))$  as a combination of the prior hidden in RNNs at all times throughout training. The pdf of words across the entire vocabulary at linear interpolation  $l+1$  may be calculated using the output  $ml$  from the memory space by using the formula  $pl+1 = z(ml) = [1]$ . Images and text are encoded into fixed-length vectors and utilized as inputs to a LSTM network. Each image's CNN characteristics are computed, and then fed into a kind of reinforcement learning. The combination of a string of words and an image in a phrase result in a novel structure. The image acts as the jumping-off point for a new career, while the words constitute the remainder. By periodically executing the recursive connect for  $l$  among  $1$  &  $L_i$ , the Short short-term memories are trained and use this new sequence. The word embed matrix, LSTM tensor, and image feature linear activation matrix are all elements of a neural model. The principal of the head line model's 3 main sub models is the visual subsystem, which employs a feature map for the picture 29 repetitions with proportions of  $29 \times 4097$  (where 29 is the greatest number of words in a caption). The second approach, a perceptron (NLP with an one LSTM unit, generates a  $28$ -by- $256$  grid, with 128 is the return dimension of the LSTM layer; the third model takes these two matrices as input and feeds data into a final LSTM unit, which has proportions of  $28$ -by- $915$ . For training, we utilize an unaltered encrypted word vector as the candidate solution; for tests, we use a separate coded text matrix and append the feature map from the test image, resulting in a matrix with size of  $29$  by  $915$ .

## V. RESULTS

A. Such sets of data include photos and textual descriptions of those images written in a natural language like English. Table I displays the metrics of sets of data. Experts in these data provide unbiased, eye-catching descriptions of each picture using a total of five phrases.

TABLE I  
DATASET STATISTICS

Dataset Name	Size		
	Train	Valid	Test
Flickr8k [1]	6000	1000	1000
Flickr30k [1]	28000	1000	1000
MSCOCO [1]	82783	40504	40775

B. Outcomes Fifty training epochs have been completed on the model. As additional eras are utilized, the loss may be reduced to 3.74. Taking the amount of data into account requires additional epochs to get reliable findings.



Fig 5 Result Of Caption Generation For Images

## VI. CONCLUSION

In this research, a model is introduced; it is a multilayer perceptron that can immediately analyze an image and provide meaningful captions in human languages including English. Program is taught to produce a word or description from an image. The categories below outline how model-derived captions and descriptions are often used. Error-free description, Minimal inaccuracies Slightly off-image description Completely off-image description. Terms like "vehicle" and "car" and "taxi" and "cabbie" and "limo" Extensive testing shows that a better best model may be achieved by using a bigger training dataset. Accuracy is improved and losses are decreased because of the increased dataset. It's also intriguing to consider how unsupervised information, in the form of both photos and texts, may be utilized to enhance current methods for generating captions for images.

## REFERENCES

- [1] Vinyals, Oriol, et al." Show and tell: A neural image caption generator." Computer Vision and Pattern Recognition (CVPR), 2021 IEEE Conference on. IEEE, 2021.
- [2] Gerber, Ralf, and N-H. Nagel." Knowledge representation for the generation of quantified natural language descriptions of vehicle traffic in image sequences." Image Processing, 1996. Proceedings., International Conference on. Vol. 2. IEEE, 2019.
- [3] Yao, Benjamin Z., et al." I2t: Image parsing to text description." Proceedings of the IEEE 98.8 (2020): 1485- 1508.
- [4] Farhadi, Ali, et al." Every picture tells a story: Generating sentences from images." Euro-pean conference on computer vision. Springer, Berlin, Heidelberg, 2022.
- [5] Yang, Yezhou, et al." Corpus-guided sentence generation of natural images." Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2018.
- [6] Kulkarni, Girish, et al." Babytalk: Understanding and generating simple image descriptions." IEEE Transactions on Pattern Analysis and Machine Intelligence 35.12 (2019): 2891-2903.
- [7] Mitchell, Margaret, et al." Midge: Generating image descriptions from computer vision de-tectons." Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, 2020.
- [8] Kuznetsova, Polina, et al." Collective generation of natural image descriptions." Proceed-ings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1. Association for Computational Linguistics, 2019.
- [9] Jia, Xu, et al." Guiding long-short term memory for image caption generation." arXiv pre-print arXiv:1509.04942 (2022).
- [10] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio." Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2018).
- [11] Mao, Junhua, et al." Deep captioning with multimodal recurrent neural networks (m-RNN)." arXiv preprint arXiv:1412.6632 (2022).
- [12] Karpathy, Andrej, and Li Fei-Fei." Deep visual- semantic alignments for generating image descriptions." Proceedings of the IEEE conference on computer vision and pattern recog-nition. 2019.
- [13] Xu, Kelvin, et al." Show, attend and tell: Neural image caption generation with visual attention." International Conference on Machine Learning. 2019.
- [14] El Housseini, Ali, Abdelmalek Toumi, and Ali Khenchaf." Deep Learning for target recognition from SAR images." Detection Systems Architectures and Technologies (DAT), Seminar on. IEEE, 2017.
- [15] Simonyan, Karen, and Andrew Zisserman." Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2020).
- [16] Donahue, Jeffrey, et al." Long-term recurrent convolutional networks for visual recognition and description." Proceedings of the IEEE conference on computer vision and pattern recognition. 2021.
- [17] Lu, Jiasen, et al." Knowing when to look: Adaptive attention via a visual sentinel for image -



captioning.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Vol. 6. 2017.

- [18] Ordonez, Vicente, Girish Kulkarni, and Tamara L. Berg.” Im2text: Describing images using 1 million of the captioned photographs.” Advances in neural information processing systems. 2019.
- [19] Chen, Xinlei, and C. Lawrence Zitnick.”Mind’s eye: A recurrent visual representation for image caption generation.” Proceedings of the IEEE conference on computer vision and pattern recognition. 2021.
- [20] Feng, Yansong, and Mirella Lapata.” How many words is a picture worth? automatic caption generation for news images.” Proceedings of the 48th annual meeting of the Association for Computational Linguistics. Association for Computational Linguistics, 2019.
- [21] Rashtchian, Cyrus, et al.” Collecting image annotations using Amazon’s Mechanical Turk.” Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and The Language Data with Amazon’s Mechanical Turk. Association for Computational Linguistics, 2020.