

# **VISUAL SEMANTIC EXTRACTION FOR TEXTUAL DESCRIPTION USING CNN AND LSTM**

A Major Project Report Submitted  
In partial fulfillment of the requirements for the award of the degree of

## **Bachelor of Technology in Information Technology**

by

<b>SAHADEV BHAGANAGARE</b>	<b>20N31A12D6</b>
<b>U. MUKESH GOPI NANDH</b>	<b>20N31A12F5</b>
<b>P. NITHIN KUMAR</b>	<b>20N31A12H9</b>

Under the esteemed guidance of

**Dr. K. Suresh**  
**Assistant Professor**



**Department of Information Technology**

**Malla Reddy College of Engineering & Technology**

(Autonomous Institution- UGC, Govt. of India)

(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA & NAAC with 'A' Grade)

Maisammaguda, Kompally, Dhulapally, Secunderabad – 500100

website: [www.mrcet.ac.in](http://www.mrcet.ac.in)

**2020-2024**



# **Malla Reddy College of Engineering & Technology**

(Autonomous Institution- UGC, Govt. of India)

(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA & NAAC with 'A' Grade)

Maisammaguda, Kompally, Dhulapally, Secunderabad – 500100

website: [www.mrcet.ac.in](http://www.mrcet.ac.in)

## **CERTIFICATE**

This is to certify that this is the bonafide record of the project entitled “Visual Semantic Extraction for Textual Description using CNN and LSTM ”, submitted by Sahadev Bhaganagare(20N31A12D6), U.Mukesh Gopi Nandh(20N31A12F5) and P. Nithin Kumar(20N31A12H9) of B.Tech in the partial fulfillment of the requirements for the degree of Bachelor of Technology in Information Technology, Department of IT during the year 2023-2024. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma.

**Internal Guide**

**Dr. K. Suresh**  
**Assistant Professor**

**Head of the Department**

**Dr. G.Sharada**  
**Professor**

**External Examiner**

## **DECLARATION**

We hereby declare that the project titled “Visual Semantic Extraction For Textual Description Using CNN and LSTM” submitted to Malla Reddy College of Engineering and Technology (UGC Autonomous), affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH) for the award of the degree of Bachelor of Technology in Information Technology is a result of original research carried-out in this report. It is further declared that the project report or any part thereof has not been previously submitted to any University or Institute for the award of degree or diploma.

**Sahadev Bhaganagare     - 20N31A12D6**

**U. Mukesh Gopi Nandh   - 20N31A12F5**

**P. Nithin Kumar         - 20N31A12H9**

## **ACKNOWLEDGEMENT**

We feel honored to place our warm salutation to our college Malla Reddy College of Engineering and Technology (UGC-Autonomous) for giving us an opportunity to do this Project as part of our B.Tech Program. We are ever grateful to our Director Dr. VSK Reddy and Principal Dr.S.Srinivasa Rao who enabled us to have experience in engineering and gain profound technical knowledge.

We express our heartiest thanks to our HOD, Dr. G. Sharada for encouraging us in every aspect of our course and helping us realize our full potential.

We would like to thank our Project Guide Dr. K. Suresh for his regular guidance, suggestions and constant encouragement. We are extremely grateful to our Project Coordinator Mrs. N. Prameela for her continuous monitoring and unflinching co-operation throughout project work.

We would like to thank our Class Incharge Mr. K. Naga Koushil Reddy who in spite of being busy with his academic duties took time to guide and keep us on the correct path.

We would also like to thank all the faculty members and supporting staff of the Department of IT and all other departments who have been helpful directly or indirectly in making our project a success.

We are extremely grateful to our parents for their blessings and prayers for the completion of our project that gave us strength to do our project.

With regards and gratitude

**Sahadev Bhaganagare - 20N31A12D6**

**U. Mukesh Gopi Nandh - 20N31A12F5**

**P. Nithin Kumar - 20N31A12H9**

## **ABSTRACT**

The process of generating a textual description for images is known as image captioning. Now a days it is one of the recent and growing research problem. Day by day various solutions are being introduced for solving the problem. Eventhough, many solutions are already available, a lot of attention is still required for getting better and precise results. So,we came up with the idea of developing an image captioning model using different combinations of Convolutional Neural Network architecture along with Long Short Term Memory in order to get better results.

This project focuses on the realm of image captioning with the help of machine learning, employing advanced neural network architectures to unite the semantics of visual content and natural language descriptions. The project “Visual semantic extraction for textual description using CNN and LSTM” is a deep learning based project that leverages the power of convolutional neural networks (CNNs) for image feature extraction and long short term memory (LSTM) for sequential language generation, our model endeavours to independently generate descriptive captions for diverse image. There are multiple use cases and applications of this model like accessibility, social media content sharing, human-computer interaction and robotics.

In conclusion, image captioning, driven by advancements in machine learning, has emerged as a powerful tool forbridging the gap between visual content and natural language. The continuous improvement in accuracy and versatilitysignifies its growing significance in diverse applications, promising a more accessible and enriched user experience across various domains.

## **TABLE OF CONTENTS**

<b><u>S.NO</u></b>	<b><u>TITLE</u></b>	<b><u>PG.NO</u></b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 PURPOSE AND OBJECTIVES	2
	1.2 EXISTING AND PROPOSED SYSTEM	3
	1.3 SCOPE OF PROJECT	6
<b>2</b>	<b>LITERATURE SURVEY</b>	<b>7</b>
<b>3</b>	<b>SYSTEM ANALYSIS</b>	<b>9</b>
	3.1 HARDWARE AND SOFTWARE REQUIREMENTS	9
	3.2 SOFTWARE REQUIREMENTS SPECIFICATION	11
<b>4</b>	<b>SYSTEM DESIGN</b>	<b>13</b>
	4.1 DESCRIPTION	13
	4.2 ARCHITECTURE	15
	4.3 UML DIAGRAMS	21
<b>5</b>	<b>METHODOLOGY</b>	<b>25</b>
	5.1 TECHNOLOGIES USED	25
	5.2 MODULES DESCRIPTION	30
	5.3 PROCESS/ ALGORITHM	34
<b>6</b>	<b>IMPLEMENTATION</b>	<b>39</b>
	6.1 SAMPLE CODE	39
	6.2 OUTPUT SCREENS	47
	6.3 TEST CASES	51
<b>7</b>	<b>CONCLUSION</b>	<b>52</b>
	<b>BIBLIOGRAPHY</b>	<b>53</b>

# **CHAPTER 1**

## **INTRODUCTION**

Visual Semantic Extraction for Textual Description is a fascinating field at the intersection of computer vision and natural language processing (NLP), aiming to generate textual descriptions for images automatically. This documentation provides an overview of using Convolutional Neural Networks (CNNs) and Long Short-Term Memory networks (LSTMs) for image captioning, outlining their architectures and how they work together to achieve this task.

There are multiple use cases and applications of this model like accessibility, social media contentsharing, human-computer interaction and robotics.

Labeling the satellite picture with atmospherical conditions and various captions of land cover or land useis challenging. The results of used algorithms will enable the worldwide community for a better understanding of what, how, and why deforestation is happening everywhere over the globe - and the ultimate way to reply. Furthermore, existing methods generally can't differentiate between man causes of forest loss and natural one. Higher resolution imagery has already been shown to be exceptionally good atthis, but robust methods haven't yet been developed for Planet imagery. To overcome this problem ouraim is developing a combination of CNN and RNN algorithm encoder decoder architecture to caption these satellite images. The data images were carried out from Earth's full frame analytic scene products using 4 class satellites in sun synchronously orbit and International artificial satellite orbit. Each contains a few bands of information: green, red, blue, infrared and therefore the set of chips for this project uses an actual pattern. The precise spectral responses of the satellites used for images are found within the Planet documentation. Each of those channels is in a 16-bit digital number format that meets the specification of the world. An inventory of training file names and their labels, the labels are space-delimited.

High resolution of images have already shown the proof of exceptionally better performance at this, but the robust methodologies haven't yet been developed for earth imagery. Overcoming this the problem our aim is developing a combination of algorithm, encoder decoder architecture to caption these satellite images. Specifically, we trained deep convolutional neural networks (CNNs) to find out image features and used multiple classification frameworks including long short-term memory (LSTM) label captioning and binary cross entropy to predict multi-class, multi-label images.

## 1.1 PURPOSE AND OBJECTIVES

The purpose Visual semantic extraction for textual description is to bridge the gap between visual information and textual understanding, making images more accessible, interpretable, and useful in a wide range of applications. Image caption generators leverage the capabilities of modern machine learning models to automatically generate descriptive text, contributing to a more comprehensive understanding of visual content.

Visual semantic extraction for textual description involves developing a system that can automatically generate descriptive captions for images. This area is at the intersection of computer vision and natural language processing, and it has several applications and potential scopes.

The objectives for an visual semantic extraction for textual description are consists of:

**Generate Descriptive Captions:** Develop a system capable of automatically generating descriptive and accurate captions for a wide variety of images.

**Improve Accessibility:** Enhance accessibility for visually impaired individuals by providing textual descriptions of images in digital content.

**Enhance Content Understanding:** Improve content understanding and user engagement by supplementing images with informative textual descriptions.

**Facilitate Image Retrieval:** Enable efficient image retrieval by associating images with relevant textual descriptions, enhancing search capabilities in image-heavy databases or applications.

**Support Multimodal Applications:** Enable integration with multimodal applications by providing a natural language interface for interacting with images.

**Explore Advanced Techniques:** Investigate and implement advanced techniques such as attention mechanisms, reinforcement learning, or multimodal fusion to improve captioning performance.



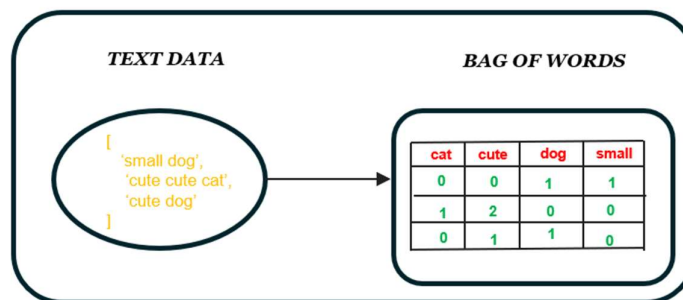
## 1.2 EXISTING AND PROPOSED SYSTEM

### ➤ Existing System

Visual semantic extraction for textual description existed, Some notable systems include Bag of Words and Nearest Neighbor, etc.

#### • Bag-of-Words

1. Involves dividing an image, assigning name to each part and considering the highest frequency name as name of image.
2. The Methodology of Bag of Words are Feature Extraction, Vocabulary Construction, Vectorization, Image Description.
3. The Limitations of Bag of Words are Lack of Control, Limited Vocabulary, Semantic Gap.



#### • Nearest Neighbor

1. Searches for an image similar to the provided image in the database and give the same caption.
2. The Methodology of Bag of Words are Feature Extraction, Nearest Neighbor Search, Caption Transfer.
3. The Limitations of Bag of Words are Limited Diversity, Semantic Gap.

	Reference	Nearest neighbors by generated features (in order left to right for images. Top to bottom for captions)
Image		  
Caption	A zebra standing in a field of grass	A zebra standing in a field of grass A group of zebras standing in a field A group of cows are standing in a field

## ➤ **Proposed System**

This project proposes a method that uses a CNN and LSTM model.

- **Convolutional Neural Network(CNN)**

Convolutional Neural Network (CNN) is a Deep Learning algorithm which takes in an input image and assign importance (learnable weights and biases) to various aspects/objects in the image, which helps it differentiate one image from the other. One of the most popular applications of this architecture is image classification. The neural network consists of several convolutional layers mixed with nonlinear and pooling layers. When the image is passed through one convolution layer, the output of the first layer becomes the input for the second layer. This process continues for all subsequent layers. After a series of convolutional, nonlinear and pooling layers, it is necessary to attach a fully connected layer. This layer takes the output information from convolutional networks. Attaching a fully connected layer to the end of the network results in an N dimensional vector, where N is the number of classes from which the model selects the desired class.

- **Long short-term Memory**

Long Short-Term Memory (LSTM) networks are a type of Recurrent Neural Network(RNN) capable of learning order dependence in sequence prediction problems. This is most used in complex problems like Machine Translation, Speech Recognition, and many more. The reason behind developing LSTM was, when we go deeper into a neural network if the gradients are very small or zero, then little to no training can take place, leading to poor predictive performance and this problem was encountered when training traditional RNNs. LSTM networks are well- suited for classifying, processing, and making predictions based on time series data since there can be lags of unknown duration between important events in a time series. LSTM is way more effective and better compared to the traditional RNN as it overcomes the short term memory limitations of the RNN. LSTM can carry out relevant information throughout the processing of inputs and discards non-relevant information with a forget gate.

- **Deep Learning**

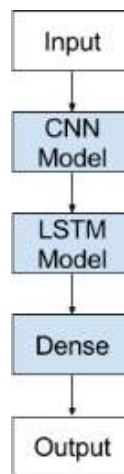
Deep Learning (DL) is a type of machine learning that uses artificial neural networks to

learn from data and improve over time. It enables machines to make decisions on their own, without being explicitly programmed, and is used for tasks such as image recognition, natural language processing, and speech recognition.

- **CNN LSTM Architecture**

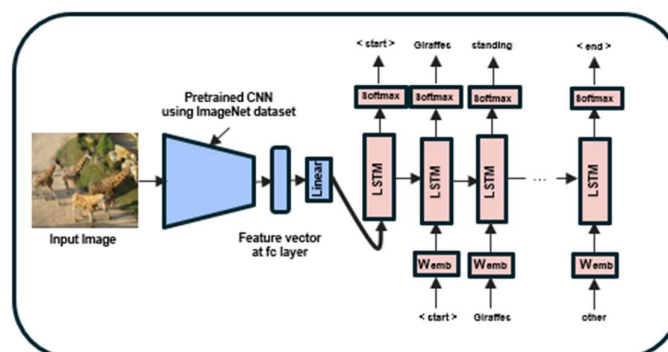
The CNN-LSTM architecture involves using CNN layers for feature extraction on input data combined with LSTMs to support sequence prediction. This model is specifically designed for sequence prediction problems with spatial inputs, like images or videos. They are widely used in Activity Recognition, Image Description, Video Description and many more.

The general architecture of the CNN-LSTM Model is shown below



- **CNN Based Encoder and Decoder**

This project uses the strategy of encoding and decoding which involves decoding any provided image with the help of pre trained CNN models like VGG16 that extracts the feature of image, encoding the features in the form of text with the help of RNN.



### **1.3 SCOPE OF THE PROJECT**

The scope of this project is Visual semantic extraction for textual description involves developing a system that can automatically generate descriptive captions for images. This area is at the intersection of computer vision and natural language processing, and it has several applications and potential scopes.

#### **Comprehensive Understanding:**

Provide readers with a comprehensive understanding of image captioning, covering both foundational concepts and advanced techniques.

#### **Practical Implementation:**

Equip readers with the knowledge and skills required to implement and train image captioning models effectively.

#### **Awareness of Applications:**

Increase awareness of the diverse applications and use cases of image captioning across different domains and industries.

#### **Promote Ethical Practices:**

Foster discussions on ethical considerations and promote responsible practices in the development and deployment of image captioning systems.

#### **Inspire Innovation:**

Inspire researchers and practitioners to explore innovative approaches and contribute to the advancement of image captioning technology.

## **CHAPTER 2**

### **LITERATURE SURVEY**

#### **Results and Implications of a study of 15 years of spatial picture distinction**

**experiments” [1].** The effort of this paper promotes the distinction of images along with the goal of creating high-quality Thematic maps with accurate creation of satellite image class. Few researches have pressed upon the betterment of the distinguishing process, another one is on the verge of using famous distinction architecture in certain kinds of remote sensing fields. The distinction is regarding a basic structure in remote sensing, that is found to be at the depth of conversion to spatial image classification.

#### **Spatial Picture Classification functions and techniques: A Review. Global journal of computer applications” [2].**

This paper focuses the spatial image distinction process that includes combining the pixel attributes of images to an appropriate class. Various picture distinction ideology or methods are present. According to this paper, the spatial image distinction functions are widely distinguished into 3 categories 1) hybrid 2) manual and 3) automatic. Widely the spatial picture class functions come under 1st class. Image distinction demands to choose the certain distinction criteria made on the needs. This paper is the field that consists a study on spatial image classification methods.

#### **Supervised the distinction of spatial pictures. Conference on Advances in Signal Processing” [3].**

Research in this paper focuses on the process of producing thematic from remote sensing of imagery for distinguishing images. Spectral bands non-analog integers are made to show spectral data. The data is made for non-analog distinguishing of pictures. In this paper, each pixel is distinguished through this spectral-data. Supervised and unsupervised are used for distinguishing images. This particular paper deals with the machine learning supervised distinguish mainly support vector machine, minimum distance, parallelepiped, and maximum likelihood. ANN distinction using a minimal training set. Comparison to conventional supervised classification.

#### **Photogrammetric Engineering and Remote Sensing.” [4].**

This paper deals with the strength of applying to NN computation to spatial image processing. The other AIM is to give a primary connecting of learning data in and normalize land area distinction outputs for conventional supervised and artificial neural net classes. ANN is trained to do land area classification of spatial clips of every dominant in the same way of supervised algorithms. This research is the base for creating applying weights for the future idea of software implications for ANN in the spatial image, earthly data preparation.

### **Unsupervised Change Detection in Satellite Images Using Principal Component Analysis and k- Means Clustering” [5].**

In this paper, they propose a noble technique for unsupervised algorithms to detect changes in multitemporal spatial images. They use PCA and k means clustering. here, the different images are parted in different times non-overlapping partitions. In this every pixel in the different picture is presented on a few-dimensional features array that is a project image data on the created Eigen vector area. The difference is acquired by the partition of the features array space into

the different unsupervised clusters using the k-means clustering technique with k value is two, after which assignment of every pixel-value to 1 of the two k mean clusters by distance formula called Euclidean method.

### **ArcGIS. What Is Image Classification? ArcGIS 10.5 Help Site” [6].**

This is software that is a full combination of needs in the multivariate to do supervised and unsupervised distinction. The distinction process is a work flow; the image distinction toolbar is created to give a suitable area to do classifications. These tools help with the flow for doing unsupervised and supervised distinction.

### **Multi label text distinction with a mixing model trained by electronic machine “[7].**

This paper focuses on a Bayesian classification where the multi-classes that consist of information are presented

by the mixing model. The supervised learning info shows which classes results for creating data, it could not indicate which classes were results for creating every word. Therefore we use electronic machines to complete this missing data, learning of both the distribution over combination parameters and word is distributed in every section's mixture part. They describe the advantages of this model and the current primary outputs.

### **A unified framework for multi-label image classification” [8].**

In this paper, they have utilized recurrent neural network to deal with captioning problem and combined it with CNN, the CNN cum RNN frame trained over a joint image and its labels embedded to characterize the relation of non- independency as well as the pics output relevance and it can be learned end to end from basic. The experimental outputs on community benchmarks data show that the given architecture acquires good prediction over the other state of the art label architectures.

### **Andrej Karpathy. Transfer Learning, 2017 [9].**

CS-231n is a deep learning class by Andrej on computer-vision with deep neural network labels as CNNs for computer recognition, it is recorded at Stanford University, the US in the Engineering.

## **CHAPTER 3**

### **SYSTEM ANALYSIS**

System requirements are the functionality that is needed by a system in order to satisfy the customer's requirements. System requirements are broad and a narrow subject that could be implemented to many items. The requirements document allows the project team to have a clear picture of what the software solution must do before selecting a vendor. Without an optimized set of future state requirements, the project team has no effective basis to choose The best system for your organization.

### **3.1 HARDWARE AND SOFTWARE REQUIREMENTS**

#### **3.1.2 HARDWARE REQUIREMENTS**

- **Processor (CPU):**  
A multi-core processor is recommended for parallelized computation. Intel Core i7 or equivalent with at least 4 cores is suitable for moderate-sized datasets and models
- **Random Access Memory (RAM):**  
Minimum 8 GB of RAM is recommended for basic experimentation. Larger datasets and more complex models may benefit from 16 GB or more.
- **Graphics Processing Unit (GPU):**  
A dedicated GPU is crucial for accelerating deep learning tasks. NVIDIA GPUs are commonly used in the deep learning community.
- **GPU Memory:**  
Deep learning models with large parameters may require GPUs with higher memory. A minimum of 4 GB VRAM is recommended, but 8 GB or more is preferable for more demanding tasks.
- **Storage:**  
SSD storage is recommended for faster data access. At least 256 GB SSD for the operating system and software. Additional storage space for datasets, model checkpoints, and experiment logs.

### 3.1.2 SOFTWARE REQUIREMENTS

- **Operating System:**

Recommended: Ubuntu 18.04 LTS or later, Windows 10, or macOS. Linux distributions are often preferred for deep learning due to better support for libraries and tools.

- **IDE:**

Choose a text editor or integrated development environment (IDE) for writing and running code.

- **Dependencies:**

Install necessary Python libraries such as NumPy, Matplotlib, Pillow, and others. Use virtual environments (e.g., virtualenv or conda) to manage dependencies.

- **Programming Language:**

Python is the primary programming language for deep learning.

- **GPU Drivers and CUDA:**

Install GPU drivers for your GPU model.



## 3.2 SOFTWARE REQUIREMENTS SPECIFICATION

### 3.2.1 FUNCTIONAL REQUIREMENTS

- **Feature Extraction:**

The model extracts salient features from the image, capturing its visual content (e.g., objects, colors, textures). This typically involves a Convolutional Neural Network (CNN) that encodes the image into a numerical representation

- **Caption Generation:**

Sentence Generation: The model generates a coherent sentence that accurately describes the image content. This often involves a Recurrent Neural Network (RNN) or a Transformer, utilizing the extracted features as input.

- **Vocabulary and Grammar:**

The model should have access to a large vocabulary and understand basic grammar rules to produce grammatically correct and meaningful sentences.

Attention Mechanism: Advanced models employ an attention mechanism that focuses on specific parts of the image relevant to the generated words.

- **Output Handling:**

Caption Length Control: The model should be able to generate captions of variable lengths, depending on the image complexity or user preference.

### 3.2.2 NON FUNCTIONAL REQUIREMENTS

- **Accuracy:**

Factual Accuracy: The captions should be factually correct, avoiding misleading or false information about the image content.

Relevance: The captions should be relevant to the image, focusing on the key elements and ignoring irrelevant details.

Object Recognition: The model should accurately identify and describe objects within the image, especially prominent or important ones.

- **Fluency:**

Grammar and Syntax: The captions should be grammatically correct and follow proper sentence structure. This enhances readability and professionalism.

Lexical Choice: The model should use appropriate vocabulary, avoiding overly technical terms or informal language, depending on the target audience and context.

Sentence Flow: The generated sentences should flow smoothly and connect coherently, forming a natural reading experience.

- **Diversity:**

Creativity and Variation: The model should avoid generating repetitive or generic captions for similar images. It should strive for variety and creativity in expressing the image content.

Style Adaptation: The model might be able to adapt its style based on user preferences or context, generating factual descriptions for news articles or poetic ones for artistic images.

Personalization (Optional): Advanced models might personalize captions based on individual user preferences or past interactions, leading to more engaging and relevant descriptions.

- **Efficiency:**

Speed: This ensures a smooth user experience without noticeable delays.

Computational Resources: This optimizes performance and cost-effectiveness.

## **CHAPTER 4**

### **SYSTEM DESIGN**

#### **4.1 DESCRIPTION**

System design is the process of creating a system's architecture, parts, and interfaces to ensure that it satisfies the needs of its users.

Thus, in order to examine the design of this project, we first go through the specifics of establishing the concept of drone detection through a few fundamental modules that would clearly describe the workings of the system that would come from the development.

#### **MACHINE LEARNING**

Machine learning (ML) is an area of artificial intelligence (AI) that enables computers to "learn" for themselves over time from training data and develop without explicit programming. Data patterns can be found by machine learning algorithms, which can then use this information to learn and develop their own predictions. Algorithms and models used for machine learning, in essence, gain experience.

In contrast, machine learning is a process that is automated and gives computers the ability to solve issues with little to no human involvement and make decisions based on prior experiences.

While machine learning and artificial intelligence are frequently used synonymously, they are actually two distinct ideas. Machine learning, a subset of AI that enables intelligent systems to autonomously learn new things from data, is what allows intelligent systems to make decisions, gain new abilities, and solve problems in a way that is similar to people. AI is the more general notion..

Machine learning methods are used in image classification to examine the existence of objects in a picture and classify it. The foundation of computer vision and image recognition is this specific problem. Machines don't examine an image in its entirety. They just examine pixel patterns or vectors while analysing a picture. The elements will subsequently be classified and given labels in accordance with the various rules that were established during algorithm configuration.

There are two types of Image Classification techniques:

If you are facing various pictures, and you only want to know whether the object in them is a cat or not, the problem you will have to handle is a binary classification. You only need to label one class of items for all images, or not to label it. The binary classification model is in charge of computing the presence or the absence of the object.

If there is more than one category, you handle a multiclass classification problem which implies the creation of multiple labels that will correspond to various objects. The machine will then predict which single object class is contained in the photographs or pictures, among a set of predefined labels.

These techniques both mainly rely on the way the reference images are labeled. The following parts of this article will give a more detailed presentation of the way image classification works.

## 4.2 ARCHITECTURE

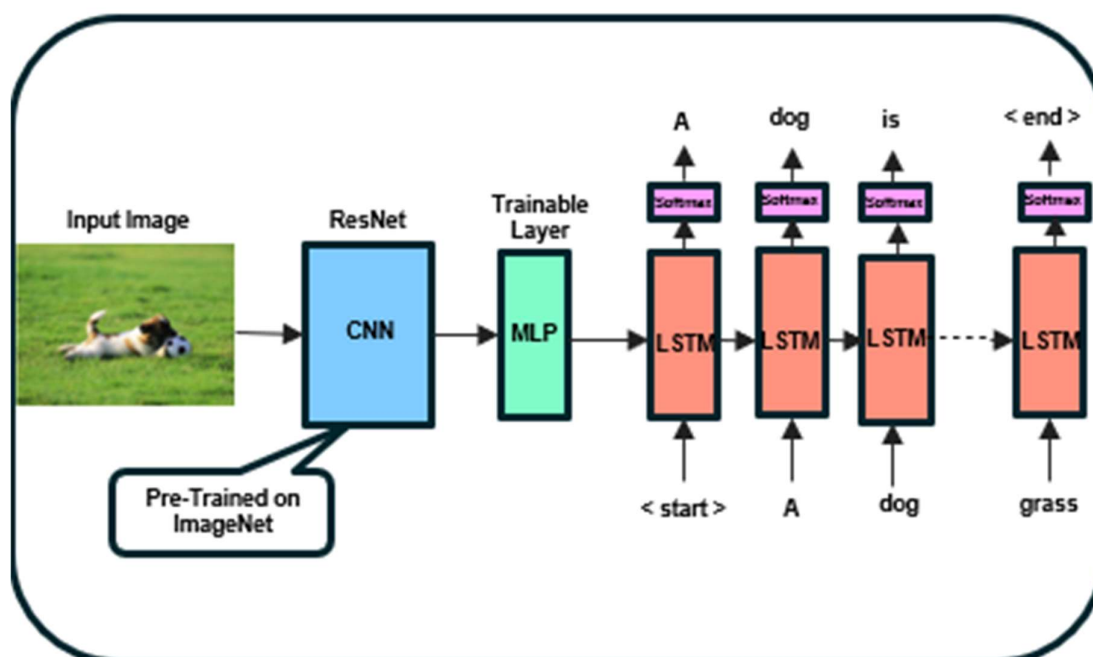
Image captioning models work like an interpreter for your eyes:

- **Decoder (CNN):**

Analyzes the image, extracting key features like shapes, colors, and textures. Thinks: "This picture has a beach, an umbrella, and people swimming."

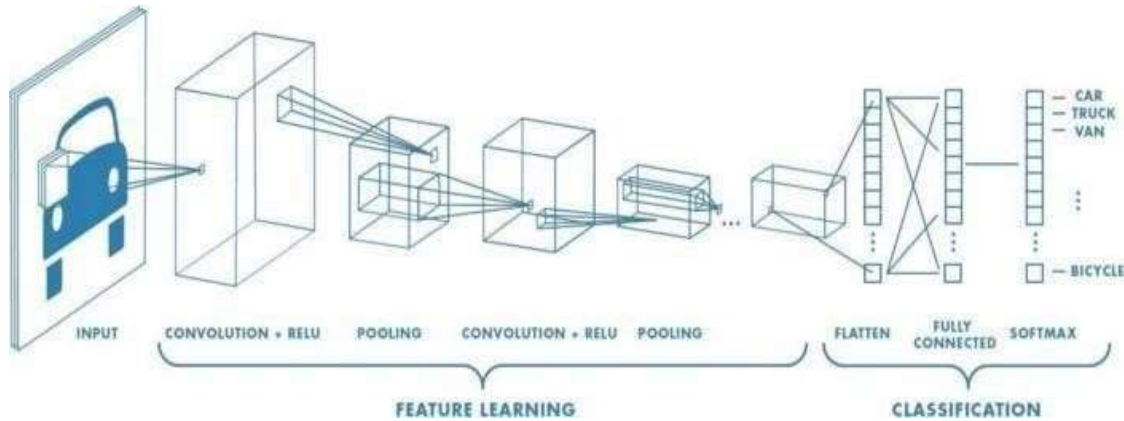
- **Encoder (RNN/LSTM):**

Uses the features to translate into words, often choosing what to focus on with "attention." Writes: "A relaxing day at the beach with friends, soaking up the sun and enjoying the cool water."



## i) CNN

Convolutional Neural Network and LSTM is being used for the development of this system.

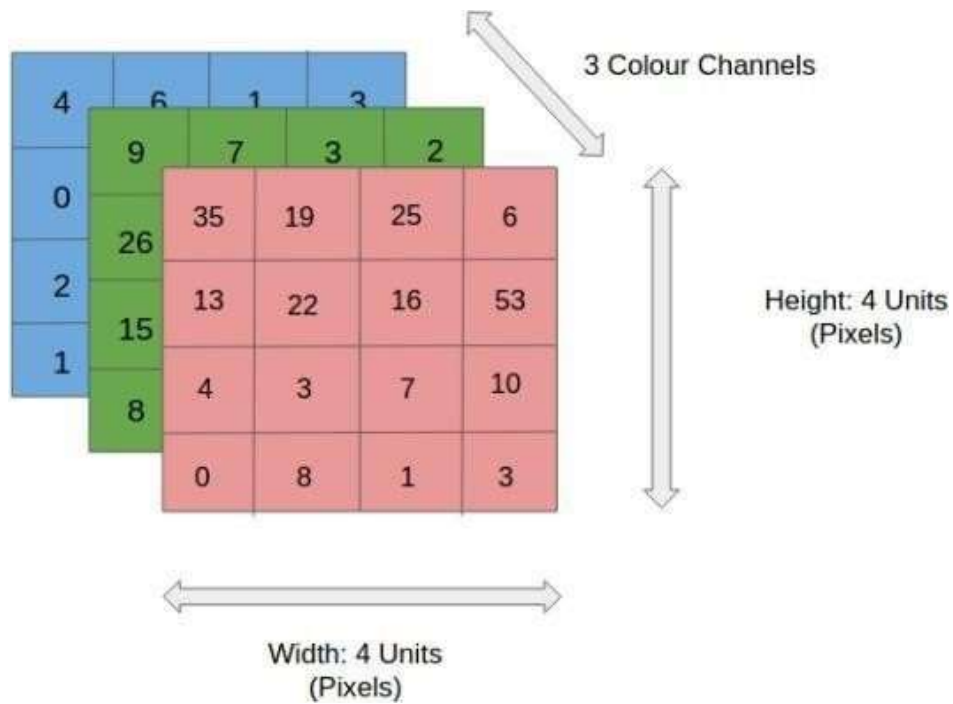


### CNN architecture image

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning method that can take in an input image, give various elements and objects in the image importance (learnable weights and biases), and be able to distinguish between them. Comparatively speaking, a ConvNet requires substantially less pre-processing than other classification techniques. ConvNets have the capacity to learn these filters and properties, whereas in primitive techniques filters are hand-engineered.

A ConvNet's architecture was influenced by how the Visual Cortex is organized and is similar to the connectivity network of neurons in the human brain. Only in this constrained area of the visual field, known as the Receptive Field, do individual neurons react to stimuli. The entire visual field is covered by a series of such fields that overlap.

A ConvNet may effectively capture the spatial and temporal dependencies in an image by using the appropriate filters. Due to the reduction in the number of parameters needed and the reuse of weights, the architecture achieves a better fitting to the picture dataset. In other words, the network may be trained to comprehend the complexity of the image more effectively.



### *Image Complexity Demonstration*

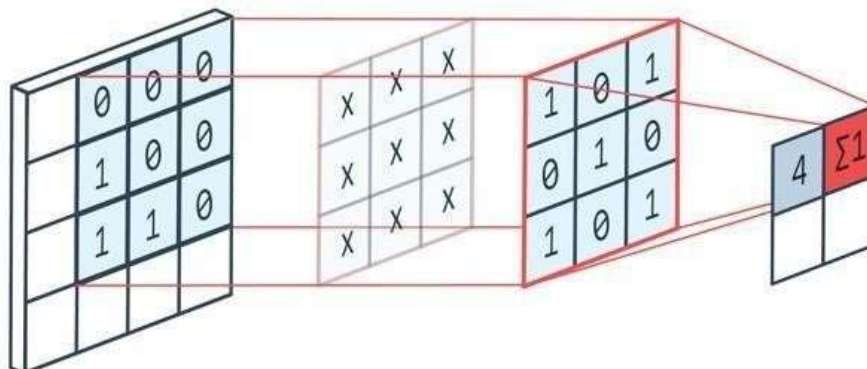
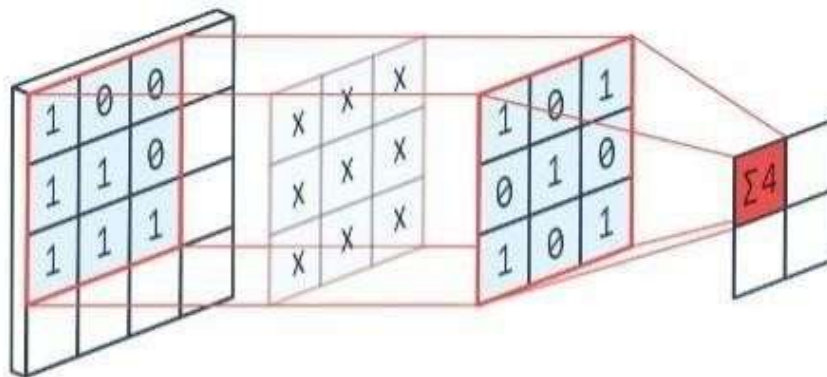
To understand the ConvNet as a whole in a simpler way, ConvNet has 3 layers in it which are listed as

1. A Convolutional Layer
2. A Pooling Layer
3. A Fully Connected Layer

## 1. A Convolutional Layer

CNN's fundamental building block is the convolution layer. The majority of the network's computational load is carried by it.

The dot product of two matrices is performed by this layer, where one matrix represents the kernel—a set of learnable parameters—the other matrix represents the constrained area of the receptive field. Compared to a picture, the kernel is smaller in space but deeper. This indicates that the kernel height and width will be spatially small if the image consists of three (RGB) channels, but the depth will go up to all three channels.

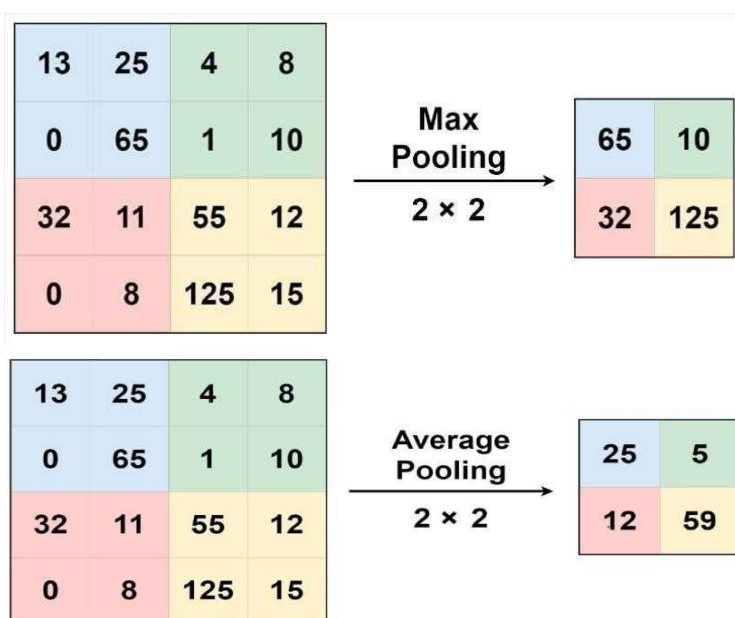




## 2. A Pooling Layer

The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

A weighted average based on the distance from the central pixel is one of the pooling functions, along with the average of the rectangular neighborhood and the L2 norm of the rectangle neighborhood. However, max pooling, which reports the highest output from the neighborhood, is the most widely used technique.



*Pooling*

## 3. A Fully Connected Layer

As with a conventional FCNN, all of the neurons in this layer are fully connected to every other neuron in the layer above and below. Because of this, it can be calculated using the standard method of matrix multiplication followed by a bias effect.

The representation between the input and the output is mapped with the aid of the FC layer.

## 1. Non-Linearity Layers

Non-linearity layers are frequently included right after the convolutional layer to add non-linearity to the activation map as convolution is a linear process and pictures are anything but linear.

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

- Sigmoid
- Tanh
- ReLU

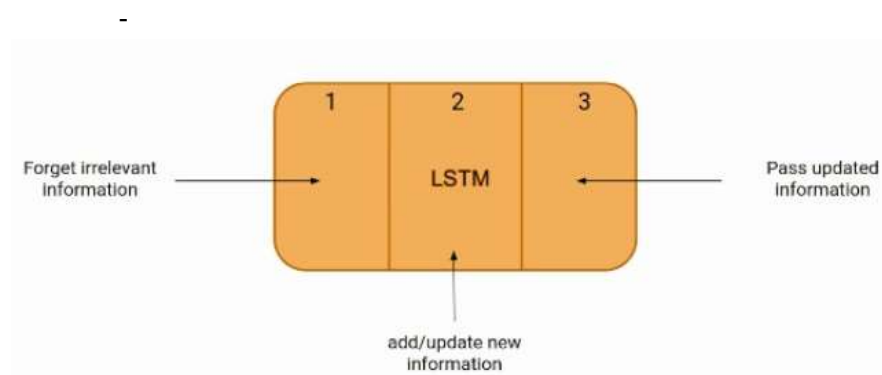
## II. LSTM

LSTM (Long Short-Term Memory) is a recurrent neural network (RNN) architecture widely used in Deep Learning. It excels at capturing long-term dependencies, making it ideal for sequence prediction tasks.

Unlike traditional neural networks, LSTM incorporates feedback connections, allowing it to process entire sequences of data, not just individual data points. This makes it highly effective in understanding and predicting patterns in sequential data like time series, text, and speech.

LSTM has become a powerful tool in artificial intelligence and deep learning, enabling breakthroughs in various fields by uncovering valuable insights from sequential data

At a high level, LSTM works very much like an RNN cell. Here is the internal functioning of the LSTM network. The LSTM network architecture consists of three parts, as shown in the image below, and each part performs an individual function.

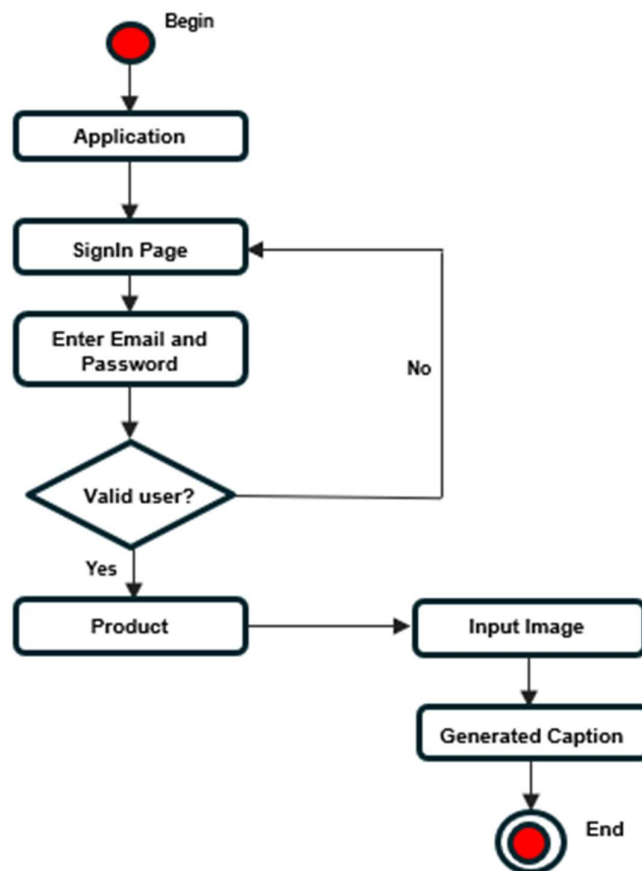


## 4.3 UML DIAGRAMS

UML Diagrams are classified into different types such as

1. ACTIVITY Diagram
2. CLASS Diagram
3. SEQUENCE Diagram

### 1. ACTIVITY Diagram



This image features a detailed activity diagram, meticulously outlining a specific process from beginning to end. At the start, the diagram introduces the application phase, swiftly moving towards a sign-in page where users are prompted to enter their email and password. The crucial decision point in this flowchart is the validation of whether the user is recognized as valid or not, leading to two distinct pathways based on the outcome.

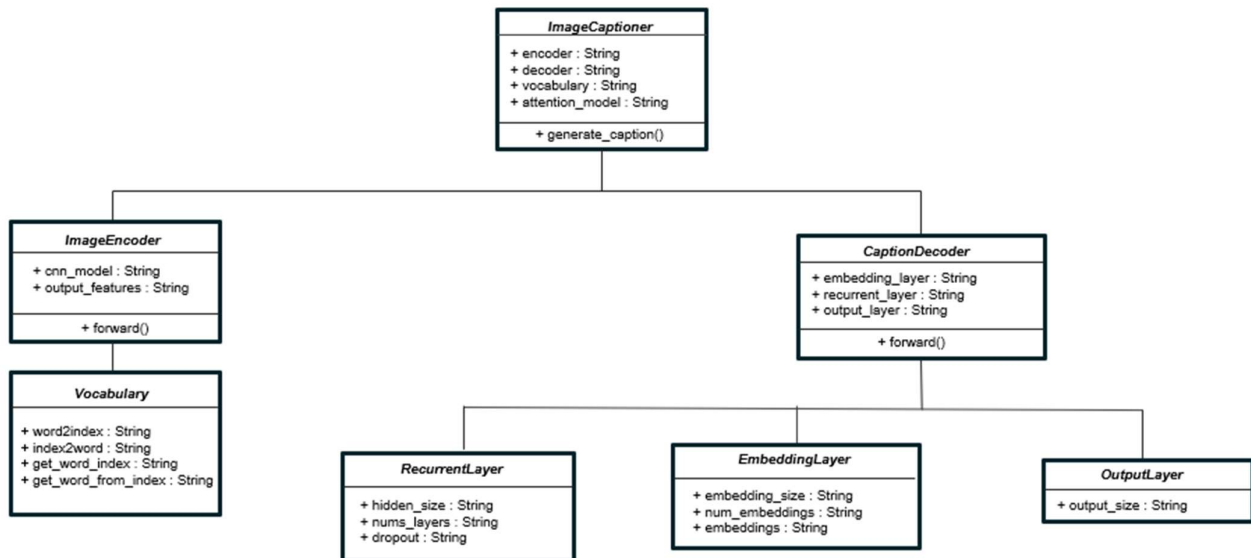
For users who are verified, the process advances towards a product stage, where an input image is processed to generate a caption, indicating an application likely dealing with image recognition or AI-based caption generation technologies. The diagram concludes with an "End" notation, signifying the completion of this process.

The visual representation is clear and concise, employing standard diagrammatic conventions to guide the viewer through the process. Text elements are prominently used to label each step, ensuring that the purpose and flow are easily understood. The use of a neutral color palette, with a dominant grey and white background, focuses attention on the content rather than the aesthetic design, making it straightforward for viewers to follow.

The inclusion of decision-making points, specifically the validation of a user, highlights the interactive and conditional nature of the process depicted. The diagram serves not only as a guide through a specific application's functionality but also illustrates the logical flow and decision-making steps involved in digital systems that require user authentication and input-based outputs.

Overall, this diagram is an informative piece that could be essential for understanding the workflow of a Software system that integrates user interaction with automated caption generation, offering insights into both the user interface and backend processing elements of the application.

## 2. CLASS Diagram



Class diagram essential for understanding the architecture of an image captioning system named "ImageCaptioner." The diagram is divided into several components, each representing a unique aspect of the system. At the core, there are two main classes: "ImageEncoder" and "CaptionDecoder," which are crucial for processing images and generating captions, respectively. The ImageEncoder class includes attributes such as a convolutional neural network (CNN) model and output features, indicating its role in transforming images into an encoded format. The CaptionDecoder class, on the other hand, is detailed with attributes like an embedding layer, recurrent layer, and output layer, highlighting its function in decoding the encoded images to generate textual captions.

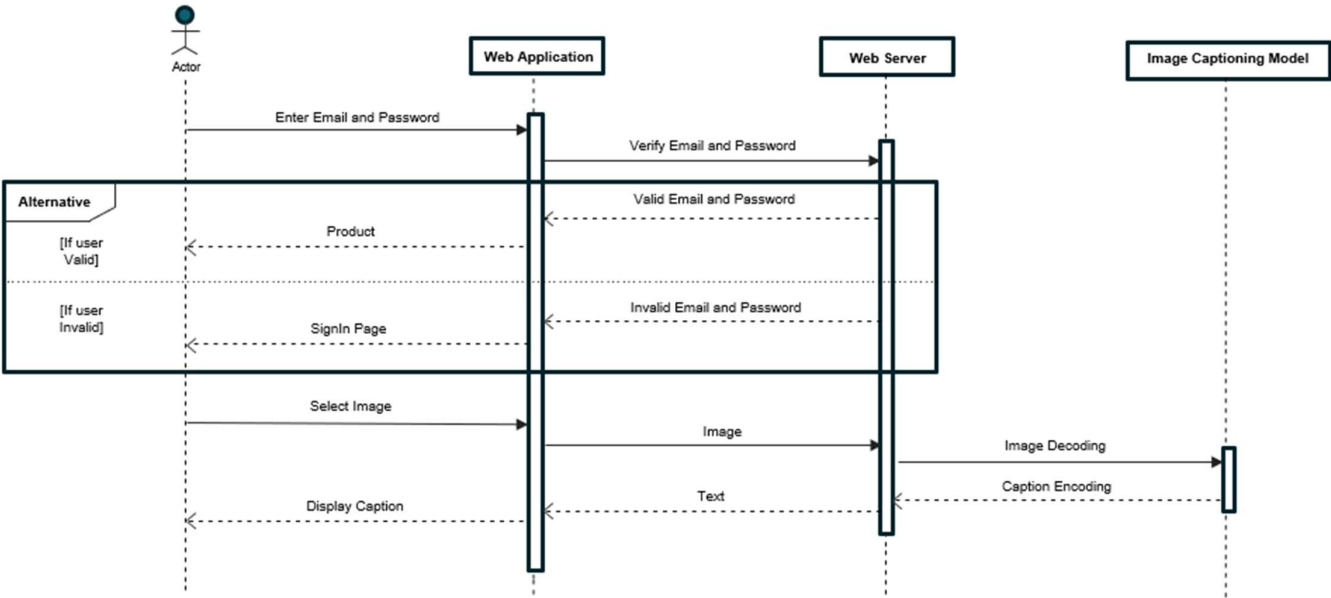
Furthermore, the diagram introduces a "Vocabulary" class with methods to map words to indexes and vice versa, facilitating the translation between textual data and numerical representations required for machine learning processes. An "EmbeddingLayer," "RecurrentLayer," and "OutputLayer" are also depicted, each playing a specific role in the caption generation process, such as handling the embeddings of words, processing sequences of data, and producing the final output captions, respectively.

Attributes across various classes include data types, all marked as `String`, which suggests that the diagram is conceptual rather than a direct software implementation. Additionally, the presence of methods like `generate_caption()` in the `ImageCaptioner` class and `forward()` in

both the Encoder and Decoder classes implies the interactive flow of data and the sequential steps involved in caption generation.

Overall, the diagram serves as an educational tool, providing a comprehensive overview of the components and processes involved in an image captioning system, highlighting the integration of image processing and natural language processing techniques to achieve the task of generating descriptive captions for images.

### 3. SEQUENCE Diagram



This a detailed sequence diagram, primarily used for illustrating the interaction and flow of processes within a web application. The diagram meticulously outlines the steps involved from an initial action by an actor, specifically entering an email and password, through the verification process, to the selection and processing of an image for captioning. It includes conditional paths indicating alternative outcomes based on the validity of the email and password provided. Notably, the diagram features distinct entities such as a Web Application, Web Server, and an Image Captioning Model, each playing a crucial role in the sequence of actions. The process flow also elaborates on image decoding and caption encoding, culminating in the display of the caption text. The design employs lines for indicating sequence flow and text annotations to describe each step and condition, ensuring clarity and ease of understanding. This visual tool is instrumental in depicting the systematic approach to handling user inputs and generating image captions, serving as an informative guide for developers and analysts interested in the inner workings of web applications and image processing models.

## **CHAPTER 5**

### **METHODOLOGY**

#### **5.1 TECHNOLOGIES USED**

##### **I. DEEP LEARNING**

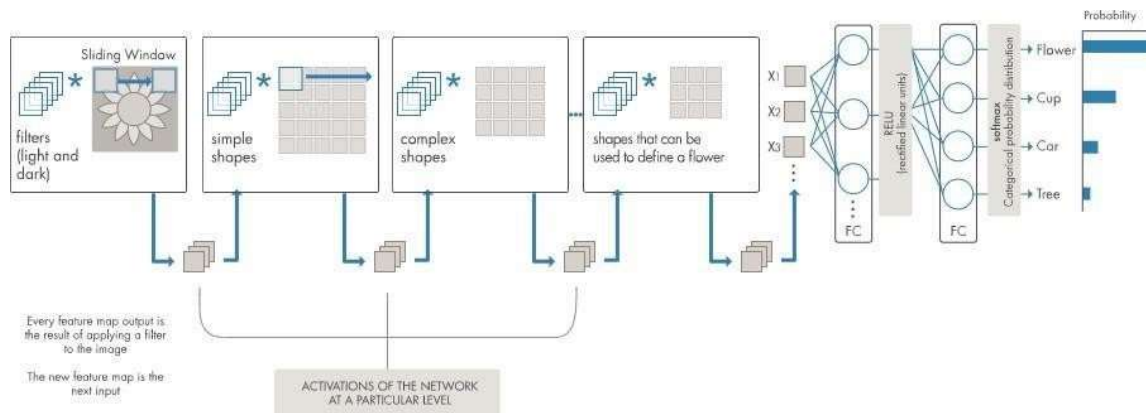
Deep learning is a machine learning method that instructs computers to learn by doing what comes naturally to people. Driverless cars use deep learning as a vital technology to recognize stop signs and tell a pedestrian from a lamppost apart. It is essential for voice control on consumer electronics including hands-free speakers, tablets, TVs, and smartphones. Recently, deep learning has attracted a lot of interest, and for good reason. It is producing outcomes that were previously unattainable.

A computer model learns to carry out categorization tasks directly from images, text, or sound using deep learning. Modern precision can be attained by deep learning models, sometimes even outperforming human ability. A sizable collection of labeled data and multi-layered neural network architectures are used to train models.

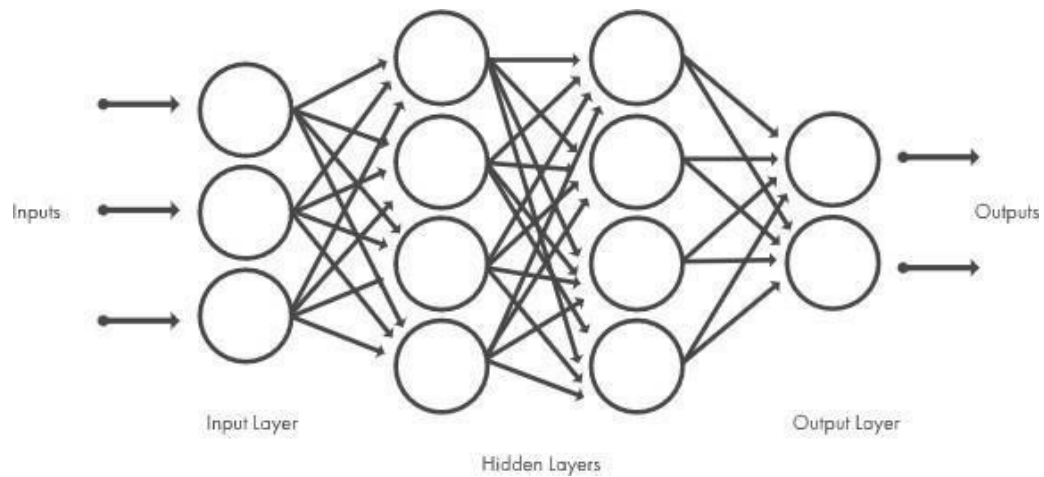
Deep learning models are sometimes referred to as deep neural networks because the majority of deep learning techniques use neural network topologies.

The number of hidden layers in the neural network is commonly referred to as "deep" in this context. Deep networks feature 150 hidden layers, compared to just 2-3 in conventional neural networks.

Large datasets of labeled data and neural network architectures that automatically extract features from the data while learning them directly from the data are used to train deep learning models



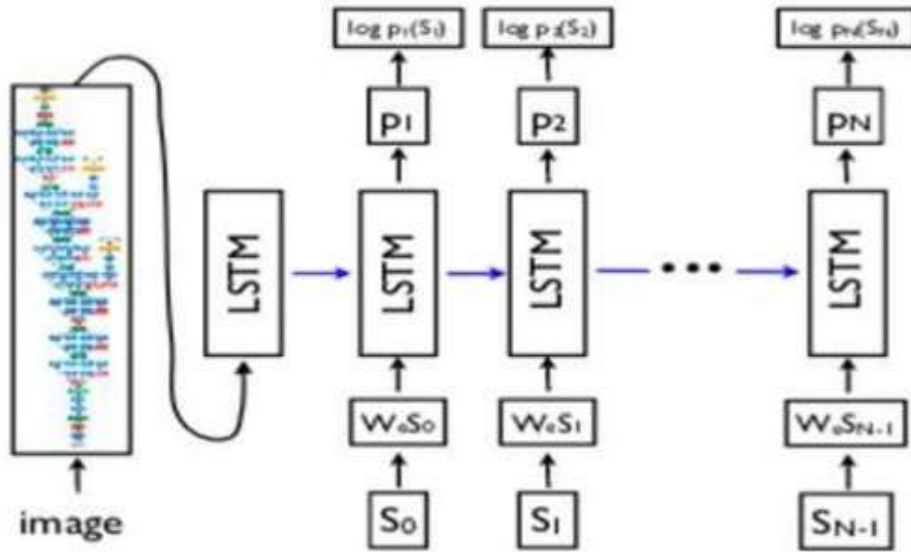
### Working of Deep Learning





## II. ENCODER DECODER ARCHITECTURE

The Encoder and Decoder architecture is utilized for a kind of setting where a variation of length of input-sequence of the sentence is mapped over the variation length provides out-sequence. The same model can also be trained for image caption or classification. In image captioning, the strong ideology is to utilize VGG19 as an encoder and a normal GRU as decoder multiple classification frameworks including long-short-term-memory (LSTM) or GRU (gated-recurrent-unit). Recurrent Neural Network is used for a variety of applications including machine language translation and chatbot model creation. The Encoder Decoder architecture is used for such practices where the varied insertion procedure is plotted over the varying length of output-array. The same network can be used under the image caption project. GRU: GRU- The Gated Recurrent Unit strives to resolve the gradient disappearing problem in the backpropagation that tags along with basic RNN. GRU is a variation on the LSTM (Long Short Term Memory) and the reason is the similar structure and, in a few instances produce similar awesome outputs in case of machine translations.



### III. MODEL CREATION AND TRAINING

#### Data Pre-Processing: Captions

In Machine Learning, data preprocessing is the key step to clean the data in order to get unified and error free data, or encode, to bring it in a certain form that the system can easily form it. To define in other way words, features, and characteristics of the data can be easily processed and interpreted by certain algorithms. One must note that the captions are the thing that everyone wants it to be predicted. So while training time, captions are the target variables or expected outputs Y that the model is training to predict. One can predict the output word by each word. Thus, we'd like to encode words in a hard and fast sized list or array. However, this part is going to be seen later once we check out the model design, except for now, we'll create two Python Dictionaries namely "word-to-ix" (pronounced as word to index) and "ix-to-word" (pronounced as an index to word). Stating that one will be representing every distinct word in created vocabulary dictionary by a number index. As seen above, we have 10000 distinct words in the dictionary and hence each word is represented by a number between 1 to 10000. The Python dictionaries are used as follows:

5.1.1 word-to-ix [„abc“] -> returns index of the word as „abc“

5.1.2 ix-to-word [p] -> returns the word whose index is as „p“

We trained deep convolutional neural network CNN to learn image features and used multiple classification frameworks including long-short-term-memory LSTM label captioning and binary cross-entropy to predict multi-class, multi-label images. Satellite images are trained on computer vision to learn image features and used classification captions including GRU label captioning and sparse\_cross\_entropy to predict multiclass, multi-label images. By fine-tuning an architecture consisting of the encoder of pre-trained VGG-19 parameters trained on ImageNet data together with the LSTM decoder.

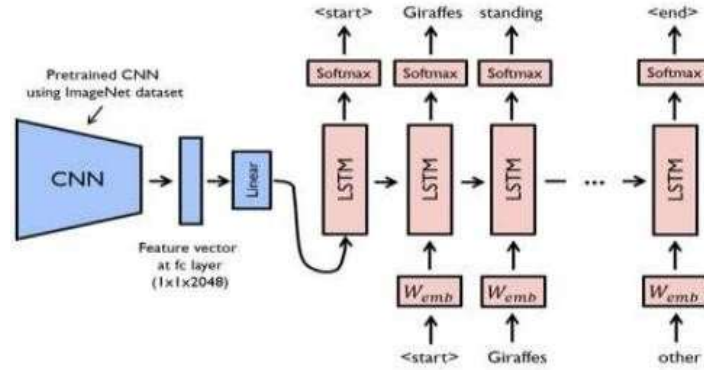
#### 4.2. Data Pre Processing: Images

Non-analog image preprocessing is the use of algorithms in order to do image pre-processing on the picture before feeding them directly to the model. As a result a subarea of digital signal processes, image process contains many merits over no analog image processor. It gives permission to a much wide angle of architecture application to the input info, the main goal of non-analog image pre- processing is or advancement of the image features by not accepting undesired distortions and promotion of few important images features so that our artificial intelligence computer Vision model can be benefited from this improved image features. Images are not a thing but input X to the encoder and decoder model. As you may already know that any X to MODEL must be given in a certain sequence of a matrix. One should transform every image into particular sized vectors that can be fed as input X to the particular net. For this to be done, one can go for transfer learning by using the VGG

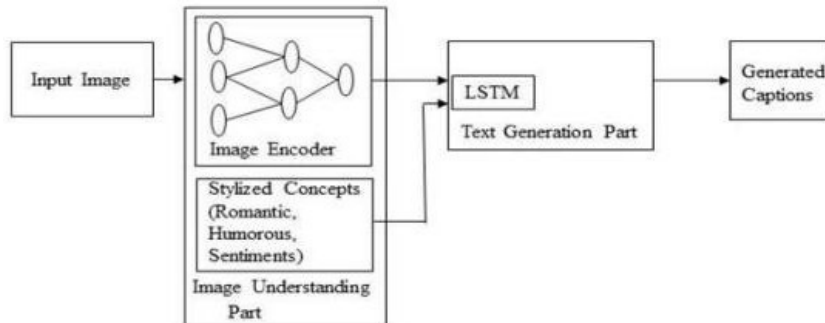
19 model Convolution Neural Network. VGG 19 was trained on Image net-datasets for image classification on thousands of different images classes. However, our purpose is to generate a caption and not to classify the image. Obtaining an informative vector for each and every picture. The process is known as feature extraction .

#### IV. TRAINED MODEL

This is the model representation of the complete architecture which show how the model is trained with some images being fed to the architectures and storing the output from the LSTM and CNN as the convolutional neural network and long shorter memory LSTM processes and stores the data of image in vector and reiterate the process until the final meaningful captions are received and also the whole images are processed.



*Training CNN and LSTM flow Diagram.*



*Block Model of implementation of encoder decoder architecture.*

## 5.2 MODULES DESCRIPTION

### MODULES

- Login
- Registration
- Image Encoding
- Caption Decoding

#### 1. LOGIN MODULE

##### a) User Authentication:

Develop a user authentication system to allow users to create accounts and log in securely.

This typically involves storing user credentials (e.g., username/email and password) securely in a database and implementing mechanisms for password hashing and salting to protect user data.

##### b) Login Page Design:

Design a login page where users can input their credentials to log into the system.

The login page should have fields for entering username/email and password, as well as options for password recovery (e.g., forgot password) if needed.

##### c) Frontend Implementation:

Use HTML, CSS, and JavaScript to create the login page's frontend.

Implement client-side validation to ensure that user inputs are properly formatted before submitting them to the server.

Handle user interactions such as form submission and error messages for invalid inputs.

##### d) Backend Implementation:

Develop the backend logic for handling login requests.

Implement server-side validation to verify user credentials against those stored in the database.

Upon successful authentication, create a session or issue a token to the user to maintain their authenticated state for subsequent requests.

## **2. REGISTRATION MODULE**

### **a) Design the Registration Page Interface:**

Design a user-friendly interface for the registration page using HTML, CSS, and possibly JavaScript.

Include input fields for users to enter their desired username, email address, password, and any additional required information.

Implement form validation to ensure that users provide valid and properly formatted information.

### **b) Backend Validation and Processing:**

Develop backend logic to handle form submissions from the registration page.

Validate user inputs on the server-side to ensure data integrity and security.

Check if the username and email address are unique and not already registered in the system.

Implement password hashing and salting techniques to securely store user passwords in the database.

### **c) Database Integration:**

Set up a database to store user registration information.

Design and create a database schema that includes tables for storing user credentials, such as usernames, email addresses, and hashed passwords.

Use a database management system (e.g., MySQL, PostgreSQL) to manage user data securely.

### **d) User Registration Logic:**

Implement backend logic to handle user registration requests.

Upon successful validation of user inputs, insert the user's information into the database.

Send a confirmation email to the user's email address for account verification, if required.

### 3. IMAGE ENCODING MODULE

#### a) Select a Pre-trained CNN Model:

Choose a pre-trained CNN model that has been trained on a large-scale image classification task such as ImageNet. Common choices include VGG16, VGG19, ResNet, Inception, or EfficientNet.

#### b) Load Pre-trained CNN Model:

Load the selected pre-trained CNN model using a deep learning framework such as TensorFlow, PyTorch.. Ensure that you include only the convolutional layers of the model, excluding the fully connected layers used.

#### c) Pre-process Images:

Pre-process the input images to match the format expected by the pre-trained CNN model.

Resize the images to the input size required by the CNN model. Normalize the pixel values to be within a certain range (typically between 0 and 1 or -1 and 1). Apply any other pre-processing steps required by the specific CNN model.

#### d) Extract Image Features:

Pass the pre-processed images through the pre-trained CNN model to extract features. Use the activations of one of the intermediate layers of the CNN as the image features. Typically, the activations from the last convolutional layer or global average pooling layer are used as feature. The output feature vectors should have a fixed length, representing the encoded features of the input images.

#### e) Dimensionality Reduction (Optional):

Optionally, apply dimensionality reduction techniques such as principal component analysis (PCA) or t-distributed stochastic neighbor embedding (t-SNE) to further compress the feature vectors if needed.

#### f) Output Encoding:

Output the encoded image features, which will be used as input to the captioning model.

The encoded image features should be in a suitable format (e.g., a tensor or numpy array) for further processing by the captioning model.

## **4. CAPTION DECODING MODULE**

### **a) Word Embedding Layer:**

Initialize a word embedding layer to convert words into dense vector representations.

Pre-trained word embeddings (e.g., Word2Vec, GloVe) can be used or learned from scratch during training.

This layer helps the model to better represent semantic similarities between words.

### **b) RNN (LSTM/GRU) Decoder:**

Define an RNN-based decoder (e.g., LSTM or GRU) to generate captions based on the encoded image features.

Initialize the hidden state of the decoder RNN using information from the final CNN layer or a learnable parameter.

At each time step, the decoder RNN takes the encoded image features and the previously generated word as input and predicts the next word in the caption.

The RNN continues generating words until an end-of-sentence token is produced or a maximum caption length is reached.

### **c) Attention Mechanism (Optional but often beneficial):**

Integrate an attention mechanism into the caption decoding module to improve the relevance and accuracy of the generated captions.

The attention mechanism allows the model to focus on different parts of the image at each time step, dynamically weighting the importance of different spatial locations.

This helps the model to generate more contextually relevant captions by attending to relevant regions of the input image adaptively.

### **d) Output Layer:**

Define an output layer to predict the probability distribution over the vocabulary at each time step.

The output layer is typically a softmax layer that computes the probabilities of each word in the vocabulary given the current hidden state of the decoder RNN.

During training, the model is optimized to maximize the likelihood of generating the ground truth caption given the input image features.

During inference, the model generates captions by sampling words from the predicted probability distributions until an end-of-sentence token is produced

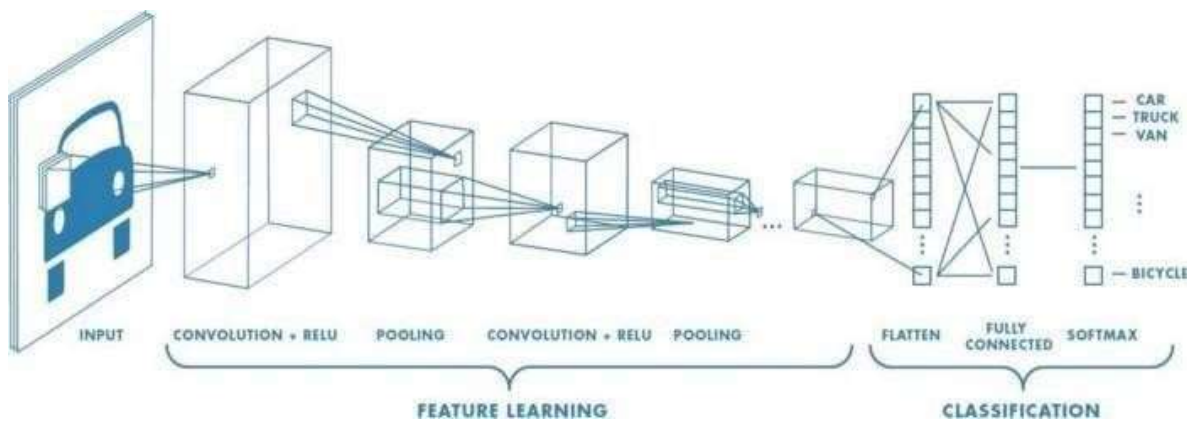
## 5.3 ALGORITHM

Algorithms used in Image Captioning are

1. CNN: For extracting the features of an image.
2. RNN: For generating text from features

### 1) CNN

Convolutional Neural Network and LSTM is being used for the development of this system.



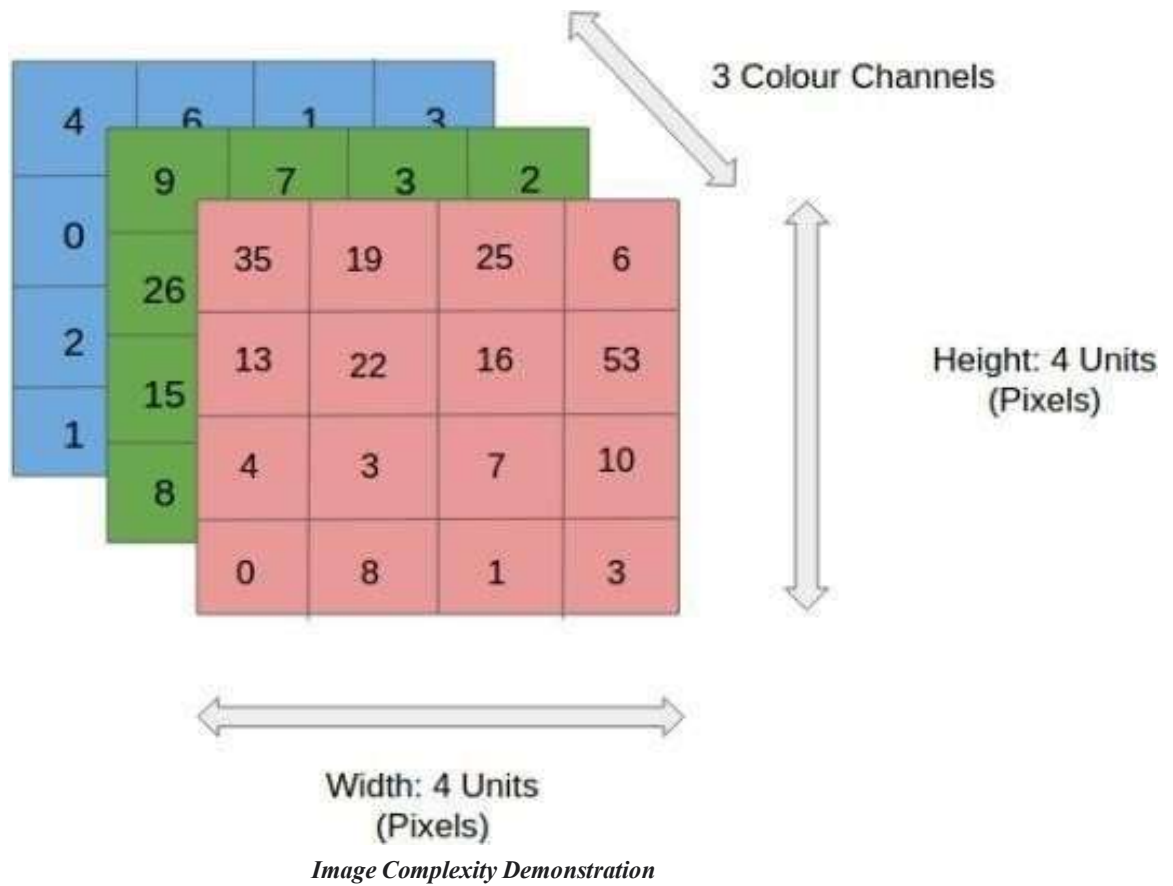
*CNN architecture image*

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning method that can take in an input image, give various elements and objects in the image importance (learnable weights and biases), and be able to distinguish between them. Comparatively speaking, a ConvNet requires substantially less pre-processing than other classification techniques. ConvNets have the capacity to learn these filters and properties, whereas in primitive techniques filters are hand-engineered.

A ConvNet's architecture was influenced by how the Visual Cortex is organized and is similar to the connectivity network of neurons in the human brain. Only in this constrained area of the visual field, known as the Receptive Field, do individual neurons react to stimuli. The entire visual field is covered by a series of such fields that overlap. A ConvNet may effectively capture the spatial and temporal dependencies in an image by using the appropriate filters. Due to the reduction in the number of parameters needed and the reuse of weights, the architecture achieves a better fitting to the picture dataset. In other words, the network may be trained to



comprehend the complexity of the image more effectively.



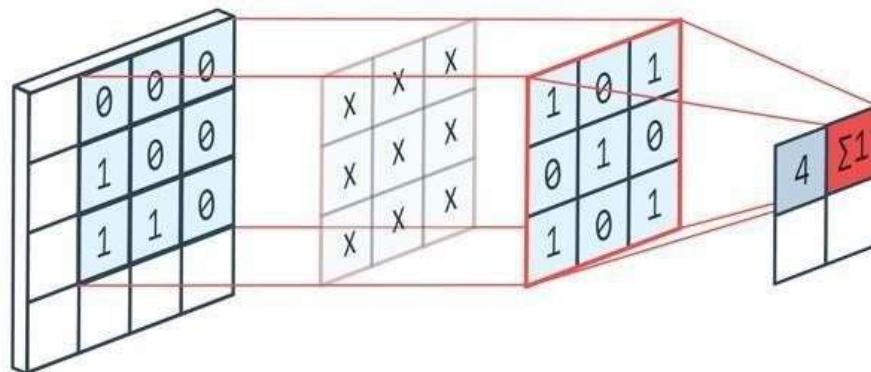
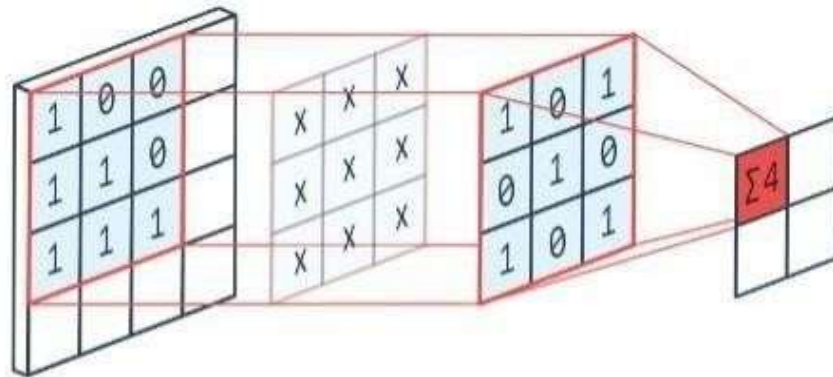
To understand the ConvNet as a whole in a simpler way, ConvNet has 3 layers in it which are listed as

1. A Convolutional Layer
2. A Pooling Layer
3. A Fully Connected Layer

## 1. A Convolutional Layer

CNN's fundamental building block is the convolution layer. The majority of the network's computational load is carried by it.

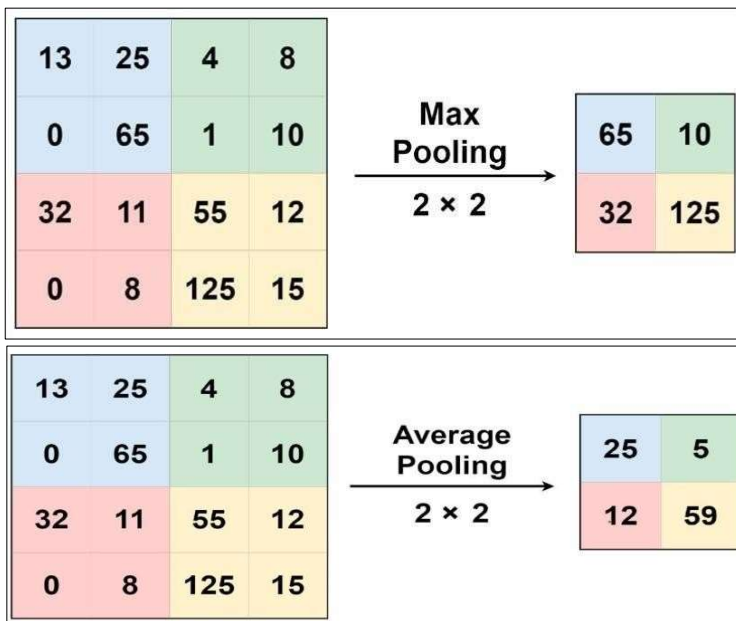
The dot product of two matrices is performed by this layer, where one matrix represents the kernel—a set of learnable parameters—the other matrix represents the constrained area of the receptive field. Compared to a picture, the kernel is smaller in space but deeper. This indicates that the kernel height and width will be spatially small if the image consists of three (RGB) channels, but the depth will go up to all three channels.



## 2. A Pooling Layer

The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

A weighted average based on the distance from the central pixel is one of the pooling functions, along with the average of the rectangular neighborhood and the L2 norm of the rectangle neighborhood. However, max pooling, which reports the highest output from the neighborhood, is the most widely used technique.



*Pooling*

## 3. A Fully Connected Layer

As with a conventional FCNN, all of the neurons in this layer are fully connected to every other neuron in the layer above and below. Because of this, it can be calculated using the standard method of matrix multiplication followed by a bias effect.

The representation between the input and the output is mapped with the aid of the FC layer.

#### 4. Non-Linearity Layers

Non-linearity layers are frequently included right after the convolutional layer to add non-linearity to the activation map as convolution is a linear process and pictures are anything but linear.

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

- i. Sigmoid
- ii. Tanh
- iii. ReLU

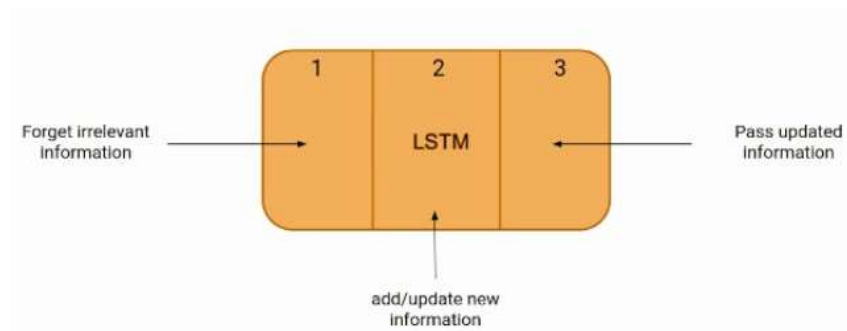
#### *II) LSTM*

LSTM (Long Short-Term Memory) is a recurrent neural network (RNN) architecture widely used in Deep Learning. It excels at capturing long-term dependencies, making it ideal for sequence prediction tasks.

Unlike traditional neural networks, LSTM incorporates feedback connections, allowing it to process entire sequences of data, not just individual data points. This makes it highly effective in understanding and predicting patterns in sequential data like time series, text, and speech.

LSTM has become a powerful tool in artificial intelligence and deep learning, enabling breakthroughs in various fields by uncovering valuable insights from sequential data

At a high level, LSTM works very much like an RNN cell. Here is the internal functioning of the LSTM network. The LSTM network architecture consists of three parts, as shown in the image below, and each part performs an individual function.



## **CHAPTER 6**

### **IMPLEMENTATION**

#### **6.1 SAMPLE OUTPUT**

##### **Server Code (app.py)**

```
from flask import Flask, render_template, redirect, request, session, url_for
import pyrebase
import subprocess
from script import generate_image_caption
import os

app = Flask(__name__)

# Firebase configuration
firebaseConfig = {
    "apiKey": "AIzaSyAxb0Y4-V3ttf2M4i2SNL-uwh0dFW-oKqg",
    "authDomain": "captioning-auth.firebaseio.com",
    "projectId": "captioning-auth",
    "storageBucket": "captioning-auth.appspot.com",
    "messagingSenderId": "820676668939",
    "appId": "1:820676668939:web:04de0a7d50ef545971610e",
    "measurementId": "G-EEB08HVN22",
    "databaseURL": "https://captioning-auth.firebaseio.com"
}

firebase = pyrebase.initialize_app(firebaseConfig)
auth = firebase.auth()

# Route for handling signup form submission
@app.route('/signup', methods=['POST'])
def signup_post():
    email = request.form['email']
    password = request.form['password']
```

```

try:
    # Create a new user with email and password
    user = auth.create_user_with_email_and_password(email, password)
    session['user'] = user['email'] # Store user email in session
    return redirect('/profile')
except Exception as e:
    error_message = str(e)
    if 'EMAIL_EXISTS' in error_message:
        session['error_message'] = 'Email already exists'
    else:
        session['error_message'] = 'An error occurred' # Generic error message
    return redirect('/signup')

# Route for login page
@app.route('/')
def login():
    error_message = session.pop('error_message', None) # Get error message from session
    return render_template('login.html', error_message=error_message)

# Route for signup page
@app.route('/signup')
def signup():
    error_message = session.pop('error_message', None) # Get error message from session
    return render_template('signup.html', error_message=error_message)

# Route for handling login form submission
@app.route('/login', methods=['POST'])
def login_post():
    email = request.form['email']
    password = request.form['password']

    try:
        # Sign in the user with email and password
        user = auth.sign_in_with_email_and_password(email, password)
        session['user'] = user['email'] # Store user email in session
        return redirect('/profile')
    except Exception as e:

```

```
session['error_message'] = 'Wrong credentials' # Store error message in session
return redirect('/')
```

```
# Route for profile page
@app.route('/profile')
def profile():
    # Check if user is logged in
    if 'user' in session:
        user_email = session['user']
        return render_template('profile.html', email=user_email)
    else:
        return redirect('/')
```

```
# Route for logout
@app.route('/logout')
def logout():
    # Clear session data to log out the user
    session.clear()
    return redirect('/')
```

```
# Route for uploading image
@app.route('/upload_image', methods=['POST'])
def upload_image():
    user_email = session['user']
    if 'image' in request.files:
        image = request.files['image']
        # Process the uploaded image (e.g., save it to disk, perform image captioning)
        # Example: Save the uploaded image to a directory
        image_path = 'uploads/' + image.filename
        print(image_path)
        image.save(image_path)

    # Run a Python script with the uploaded image as input
    image_caption = generate_image_caption(image_path)
```

```
# Redirect to profile page and pass the output to be displayed
```

```
        return render_template('profile.html', image_caption=image_caption,
                               image_path=os.path.abspath(image_path), email=user_email)
    return redirect('/profile')
```

```
if __name__ == '__main__':
    app.secret_key = 'supersecretkey'
    app.run(debug=True)
```

### **Description for app.py (Server code)**

This Python script utilizes the Flask framework to create a web application with various routes for user authentication and image uploading. Below is a description of each component and its functionality:

#### **1. \*Imports:\***

- Flask: Importing the Flask class to create the web application.
- render\_template, redirect, request, session, url\_for: Modules from Flask for rendering templates, handling redirects, managing HTTP requests, maintaining session data, and generating URLs.
- pyrebase: Module for interacting with Firebase services.
- subprocess: Module for spawning new processes.
- os: Module for interacting with the operating system.

#### **2. \*Firebase Configuration:\***

- Firebase configuration details are stored in a dictionary named firebaseConfig.

#### **3. \*Flask App Initialization:\***

- An instance of the Flask class is created with Flask(\_\_name\_\_), naming the Flask app as app.

#### **4. \*Routes:\***

- **\*\*Signup Form Submission Route (/signup):\*\***
  - Handles POST requests for user signup.
  - Retrieves email and password from the form data.
  - Attempts to create a new user with the provided email and password using Firebase authentication.



- Stores the user's email in the session and redirects to the profile page upon successful signup. If an error occurs, it redirects to the signup page with an appropriate error message.

- **\*\*Login Page Route (/):\*\***

- Renders the login page template (login.html).
- Retrieves any error message stored in the session and passes it to the template.

- **\*\*Signup Page Route (/signup):\*\***

- Renders the signup page template (signup.html).
- Retrieves any error message stored in the session and passes it to the template.

- **\*\*Login Form Submission Route (/login):\*\***

- Handles POST requests for user login.
- Retrieves email and password from the form data.
- Attempts to sign in the user with the provided email and password using Firebase authentication.

- Stores the user's email in the session and redirects to the profile page upon successful login. If an error occurs, it redirects to the login page with an appropriate error message.

- **\*\*Profile Page Route (/profile):\*\***

- Renders the profile page template (profile.html).
- Checks if the user is logged in by verifying the presence of the user's email in the session.
- Passes the user's email to the template for display. If the user is not logged in, it redirects to the login page.

- **\*\*Logout Route (/logout):\*\***

- Clears session data to log out the user and redirects to the homepage.

- **\*\*Image Upload Route (/upload\_image):\*\***

- Handles POST requests for uploading images.
- Retrieves the user's email from the session.
- Processes the uploaded image (e.g., saves it to disk, performs image captioning).
- Redirects to the profile page and passes the image caption and path for display. If no image is uploaded, it redirects to the profile page.

## 5. **\*Main Block:\***

- Sets a secret key for the app.
- Runs the app in debug mode if the script is executed directly.

Overall, this script creates a Flask web application for user authentication and image uploading, leveraging Firebase for authentication and incorporating image processing functionality.

### **Main code (Script.py)**

```
import torch
from transformers import VisionEncoderDecoderModel, ViTImageProcessor, AutoTokenizer
from PIL import Image

def generate_image_caption(image_path):
    """
    Generates a caption for a given image using a pre-trained Transformer model.

    Args:
        image_path (str): The path to the image file.

    Returns:
        str: The generated caption for the image, or an error message if caption generation fails.
    """

    # Load pretrained model, tokenizer, and feature extractor
    model = VisionEncoderDecoderModel.from_pretrained("nlpconnect/vit-gpt2-image-captioning")
    feature_extractor = ViTImageProcessor.from_pretrained("nlpconnect/vit-gpt2-image-captioning")
    tokenizer = AutoTokenizer.from_pretrained("nlpconnect/vit-gpt2-image-captioning")

    # Check for GPU availability and move model to GPU if possible
    device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
    model.to(device)

    # Set parameters for caption generation
    max_length = 16 # Maximum length of the generated caption
    num_beams = 4 # Number of beams for beam search decoding
    gen_kwargs = {"max_length": max_length, "num_beams": num_beams}

    try:
```

```

# Open the image and ensure it's in RGB mode
with Image.open(image_path) as img:
    if img.mode != "RGB":
        img = img.convert(mode="RGB")

    # Extract image features using the feature extractor
    pixel_values = feature_extractor(images=img, return_tensors="pt").pixel_values
    pixel_values = pixel_values.to(device)

    # Generate caption using the model
    with torch.no_grad():
        output_ids = model.generate(pixel_values, **gen_kwargs)

    # Decode the generated token IDs into text
    preds = tokenizer.batch_decode(output_ids, skip_special_tokens=True)
    preds = [pred.strip() for pred in preds] # Remove any leading/trailing spaces

    # Return the first generated caption or an appropriate message
    image_caption = preds[0] if preds else "No caption generated"
    return image_caption

except FileNotFoundError:
    return f'Error: Image file '{image_path}' not found.'
except OSError as e:
    return f'Error processing image '{image_path}': {e.strerror}'
except Exception as e:
    return f'Error processing image '{image_path}': {e}'
finally:
    # Clean up resources if necessary (e.g., close file handles)
    Pass

```

## Description for Script.py (Main Code)

This Python script utilizes the Transformers library to generate captions for images using a pre-trained Vision Encoder-Decoder model. Here's a breakdown of its functionalities:

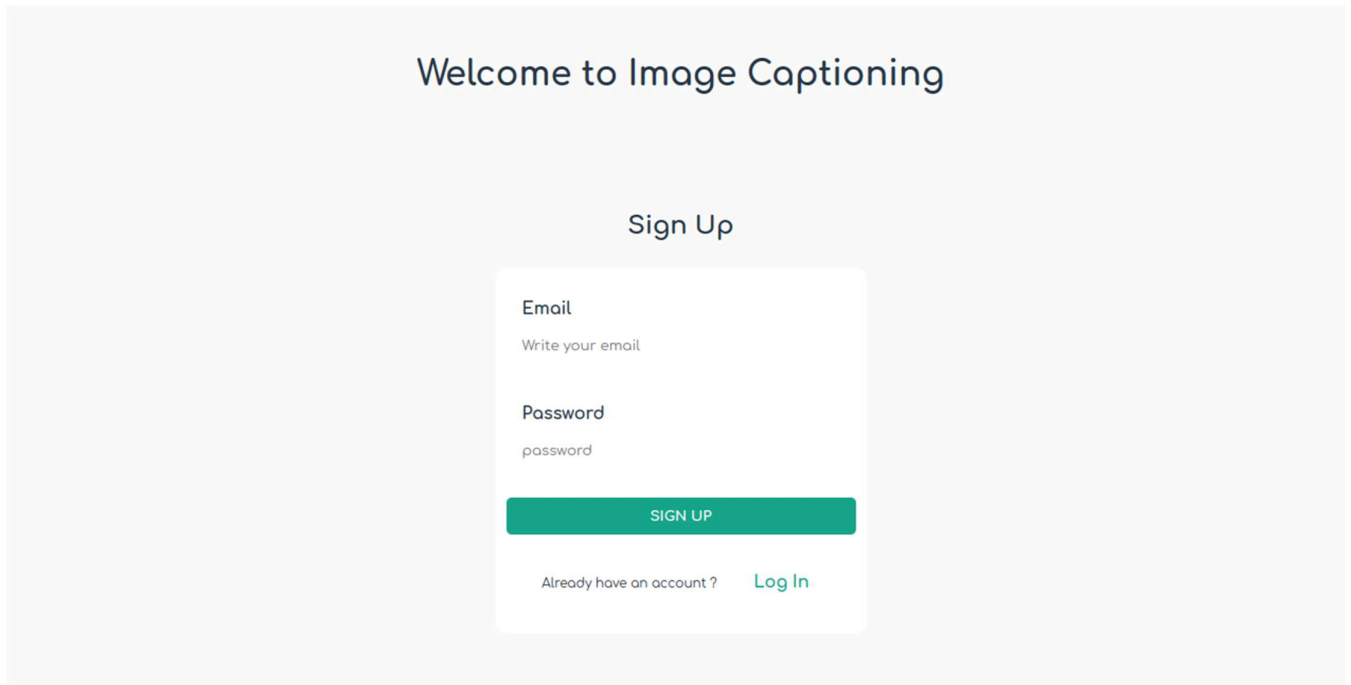
### 1. **\*\*Function generate\_image\_caption(image\_path):\*\***

- This function takes the path to an image file as input.
- It loads a pre-trained Vision Encoder-Decoder model, tokenizer, and feature extractor from the "nlpconnect/vit-gpt2-image-captioning" checkpoint.
- It checks for GPU availability and moves the model to the GPU if possible.
- Parameters for caption generation such as maximum caption length and number of beams for beam search decoding are set.
- The function opens the image file, converts it to RGB mode if necessary, and extracts image features using the feature extractor.
- It generates captions for the image using the model and decodes the token IDs into text.
- The first generated caption is returned if available; otherwise, an appropriate error message is returned.
- Error handling is implemented to catch file-related errors and other exceptions during image processing.
- Resources are cleaned up in the finally block, though there are no specific cleanup operations in this case.

This script enables automated generation of captions for images, leveraging powerful Transformer-based models, and handles various error scenarios gracefully to ensure robustness in caption generation.

## 6.2 OUTPUT SCREENS

- **REGISTRATION**



The image shows a registration form titled "Welcome to Image Captioning" and "Sign Up". The form is centered on a light gray background. It contains two input fields: "Email" with the placeholder text "Write your email" and "Password" with the placeholder text "password". Below these fields is a green "SIGN UP" button. At the bottom of the form, there is a link that says "Already have an account ? Log In", where "Log In" is in green text.

Welcome to Image Captioning

Sign Up

Email  
Write your email

Password  
password

SIGN UP

Already have an account ? [Log In](#)

- **LOGIN**

Welcome to Image Captioning

Sign Up

Email

Write your email

Password

password

SIGN UP

Already have an account ? [Log In](#)


## DASHBOARD

Image Cap

Logout mukeshudotha7@gmail.com

### Visual Semantic Extraction for Textual Description

Choose an Image...



Select any image

Limit 2000 MB per file. JPG, PNG, JPEG

Choose File

Generate Caption

No image uploaded yet.

Input 1:



Output 1:

Caption: A passport size photograph of a young man with short black hair and a beard. He is wearing a red and black plaid shirt and has a serious expression on his face.

Tags: Generated 5 hash tags: #portrait #photography #headshot #professional #profilepic

Input 2:



Output 2:

Caption: A girl is flying a kite on the terrace. She is wearing a pink shirt and has long black hair. The kite is flying high in the sky.

Tags: #kite #girl #sky #childhood #fun



### 6.3 TEST CASE

Test_ID	Test case	Steps	Expected Outcome	Actual Outcome	Status
1	Test user registrationpage	Open applicationusing local host and enter required details to register	The registration page is opened	The registration page is opened	Pass
2	Test user login page	Open login page and enter ID, password to login	The login page is opened	The login page is opened	Pass
3	Check Login credentials	Check if the usercredentials are correct	User credentials are valid	User credentials are valid	Pass
4	Taking image as input from the user	1. User selects image by clicking on choose file	User made avalid input	User made avalid input	Pass
5	User uploads image	User uploads animage Image is displayedon screen	Image is uploaded and displayed on screen	Image is uploaded and displayed on screen	Pass
6	Generate caption	1. Caption generation	Caption is successfully generated	Caption is successfully generated	Pass

## **CHAPTER 7**

### **CONCLUSION**

In this project, 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. So, to make an image caption generator model, we have merged 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. Convolutional Neural networks are specialized deep neural networks which can process the data that has input shape like a 2D matrix. Images are easily represented as a 2D matrix and CNN is very useful in working with images. CNN is basically used for image classifications and identifying if an image is a bird, a plane or Superman, etc. It scans images from left to right and top to bottom to pull out important features from the image and combines the feature to classify images. It can handle the images that have been translated, rotated, scaled and changes in perspective.

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.

## **BIBLIOGRAPHY**

- [1] G.G. Wilkinson. "Results and implications of a study of fifteen years of satellite image classification experiments." IEEE Transactions on Geoscience and Remote Sensing (Vol. 43, No. 3). 2005.
- [2] Sunitha Abburu, Suresh Babu Golla. "Satellite Image Classification Methods and Techniques: A Review". International Journal of Computer Applications, (Vol.119, No. 8). 2015.
- [3] Sayali Jog, Mrudul Dixit. "Supervised classification of satellite images". Conference on Advances in Signal Processing (CASP), 2016.
- [4] George F. Hepner. "Artificial neural network classification using a minimal training set. Comparison to conventional supervised classification". Photogrammetric Engineering and Remote Sensing, (Vol. 56, No. 4). 1990.
- [5] Turgay Celik. "Unsupervised Change Detection in Satellite Images Using Principal Component Analysis and k-Means Clustering". IEEE Geoscience and Remote Sensing Letters, (Vol. 6, No. 4). 2009.
- [6] ArcGIS. "What Is Image Classification?" "ArcGIS 10.5 Help Site, 2017.
- [7] A. McCallum. "Multi-label text classification with a mixture model trained by EM. AAAI99", Workshop on Text Learning. 1999.
- [8] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, Wei Xu. "CNN RNN: A Unified Framework for Multi-Label Image Classification". The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2285-2294.
- [9] Andrej Karpathy. Transfer Learning, 2017. CS231n: Transfer Learning.
- [10] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna. "Rethinking the Inception Architecture for Computer Vision". Computer Vision and Pattern Recognition. 2015.