

# **Image Caption Generator**

*An Industry oriented Mini Project report submitted in partial  
fulfilments of requirements for the award of degree of*

## **BACHELOR OF TECHNOLOGY**

IN

## **COMPUTER SCIENCE AND ENGINEERING**

By

**D SIVA GANESH (19131A0546)**

**CH SYAM KUMAR (19131A0543)**

**D JAYADITHYA (19131A0554)**

**G V SAI YASHWANT (19131A0565)**



**COLLEGE OF ENGINEERING  
(AUTONOMOUS)**

Under the esteemed guidance of

**Ms. Lateefa Shaik  
(Assistant Professor)**

**Department of Computer Science and Engineering**

**GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING (AUTONOMOUS)**

(Affiliated to JNTU-K, Kakinada)

**VISAKHAPATNAM**

**2021 – 2022**

**Gayatri Vidya Parishad College of Engineering (Autonomous)  
Visakhapatnam**



**COLLEGE OF ENGINEERING  
(AUTONOMOUS)**

**CERTIFICATE**

This report on  
***“Image Caption Generator”***  
is a bonafide record of the mini project work submitted

By

**D SIVA GANESH (19131A0546)**

**CH SYAM KUMAR (19131A0543)**

**D JAYADITHYA (19131A0554)**

**G V SAI YASHWANT (19131A0565)**

in their V semester in partial fulfilment of the requirements for the Award of Degree of

**Bachelor of Technology in  
Computer Science and Engineering**

During the academic year 2021-2022

**Ms. Lateefa Shaik**  
Assistant Professor  
Project Guide

**Dr.D.N.D. HARINI**  
Head of the Department  
Computer Science and Engineering

## **DECLARATION**

We hereby declare that this Socially Relevant Project entitled “**IMAGE CAPTION GENERATOR**” is a bonafide work done by us and submitted to the **Department of Computer Science and Engineering, GVP College of Engineering (Autonomous) Visakhapatnam**, in partial fulfillment for the award of the degree of B. Tech is of our own and it is not submitted to any other university or has been published any time before.

PLACE: **VISAKHPATNAM**

**D SIVA GANESH (19131A0546)**

DATE:

**CH SYAM KUMAR (19131A0543)**

**D JAYADITHYA (19131A0554)**

**G V SAI YASHWANT (19131A0565)**

## **ACKNOWLEDGEMENT**

We would like to express our deep sense of gratitude to our esteemed institute **Gayatri Vidya Parishad College of Engineering (Autonomous)**, which has provided us an opportunity to fulfil our cherished desire.

We express our sincere thanks to our Principal **Dr. A.B. KOTESWARA RAO, Gayatri Vidya Parishad College of Engineering (Autonomous)** for his encouragement to us during this project, giving us a chance to explore and learn new technologies in the form of mini project.

We express our deep sense of Gratitude to **Dr. D.N.D. HARINI, Associate Professor and Head of the Department of Computer science and Engineering, Gayatri Vidya Parishad College of Engineering (Autonomous)** for giving us an opportunity to do the project in college.

We express our profound gratitude and our deep indebtedness to our guide **Ms. LATEEFA SHAIK Assistant Professor, Department of Computer Science and Engineering**, whose valuable suggestions, guidance and comprehensive assistance helped us a lot in realizing my present project.

We also thank our coordinator, **Dr. CH SITA KUMARI, Sr. Assistant Professor, Department of Computer Science and Engineering**, for the kind suggestions and guidance for the successful completion of our project work.

Finally, we would also like to thank all the members of the teaching and non-teaching staff of the Computer Science and Engineering Department for all their support in completion of our project.

**D SIVA GANESH (19131A0546)**

**CH SYAM KUMAR (19131A0543)**

**D JAYADITHYA (19131A0554)**

**G V SAI YASHWANT (19131A0565)**

## **ABSTRACT**

Deep learning approaches have recently seen a lot of success in two disciplines: Convolutional Neural Networks (CNN) and Natural Language Processing (NLP). Analyzing the situation in an image can be done by generating information about the image. This project proposes a system to seek information about the image by generating caption to the image. Initially image(input) will be passed to the CNN model, the features obtained from CNN model will be fed to LSTM (Long-Short-Term-Memory) model and then it generates caption for the image. Generating a caption to an image can be implemented by using CNN model and the image gets converted into a fixed size vector. After pre- processing of image features like shape, colors etc. will be extracted using CNN. The features obtained from CNN model will be fed to LSTM and it generates a caption. To develop a user-friendly interface, they can generate captions to an image by using CNN-LSTM Architecture.

## **INDEX**

<b>CHAPTER 1. INTRODUCTION .....</b>	<b>8</b>
1.1 Objective.....	8
1.2 About the Algorithm.....	8
1.3 Purpose.....	12
1.4 Scope.....	13
 <b>CHAPTER 2. SRS DOCUMENT .....</b>	 <b>13</b>
2.1. Functional Requirements.....	13
2.2. Non-Functional Requirements.....	14
2.3. Minimum Hardware Requirements.....	14
2.4. Minimum Software Requirements.....	14
 <b>CHAPTER 3. ALGORITHM ANALYSIS.....</b>	 <b>15</b>
3.1. Existing Algorithm .....	15
3.2. Proposed Algorithm.....	15
3.3. Feasibility Study.....	16
3.4. Cost Benefit Analysis.....	18
 <b>CHAPTER 4. SOFTWARE DESCRIPTION.....</b>	 <b>19</b>
4.1. Anaconda Navigator... ..	19
4.2. Flask.....	19
4.3. Keras.....	19
4.4. Tensor Flow.....	19
4.5. Pillow.....	20
4.6. Matplotlib.....	20
4.7 Numpy.....	20

<b>CHAPTER 5. PROJECT DESCRIPTION .....</b>	<b>20</b>
5.1. Problem Definition.....	20
5.2. Project Overview.....	21
5.3. Module Description.....	21
5.3.1. Flask Framework.....	22
5.3.2. Problem Definition.....	24
<b>CHAPTER 6. DEVELOPMENT.....</b>	<b>31</b>
6.1. Data Sets Used.....	31
6.2. Sample Code.....	31
6.3. Results.....	39
<b>CHAPTER 7. TESTING .....</b>	<b>39</b>
7.1. Introduction of Testing.....	39
7.2. Test Code.....	43
7.3. Test Cases.....	46
<b>CHAPTER 8. CONCLUSION.....</b>	<b>47</b>
REFERENCE LINKS... ..	47

## INTRODUCTION

The process of creating a textual explanation for a set of photos is known as image captioning. In the Deep Learning arena, it has been a critical and basic task. Captioning images has a wide range of applications. Image captioning can be thought of as an end-to-end Sequence to Sequence challenge since it turns images from a series of pixels to a series of words. Both the language or remarks as well as the images must be processed for this purpose. To acquire the feature vectors in the Language part, we utilize recurrent Neural Networks (RNN) or Long Short-Term Memory (LSTM), and in the Image part, we use Convolutional Neural Networks (CNN).

### 1.1 OBJECTIVE

“IMAGE CAPTION GENERATOR” helps us in identifying the situation happening in an image. The process of analyzing is done by CNN (Convolution Neural Networks) LSTM (Long Short-Term Memory) algorithms. CNNs are deep learning machines that were originally proposed by “Yann Lecun” in 1984. Le Net is the first successful application of CNN’s. The CNN framework is designed to implement image classification, so it is the best algorithm for Image Processing. CNN is well known for its widely used applications of image and video recognition. In image classification, feature maps are extracted through convolution layers and other processing layers repetitively and the network eventually outputs a label indicating an estimated class.

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.



## 1.2 ABOUT THE ALGORITHM

Convolution Neural Network (CNN) is the algorithm used in this project for analyzing the situation in the image. We have used Alex net architecture. In Deep learning CNN is a class of artificial neural networks, Most commonly applied to analyze visual imagery. CNN works by extracting features from images. A Convolutional Neural Network is a Deep Learning algorithm which can take in an input image, assigns importance (learnable weights and biases) to various aspects/objects in the image and then makes them to differentiate from one other. Given a training dataset then CNN, unlike traditional machine learning techniques, optimizes the weights and filter parameters in the hidden layers to generate features suitable to solve the classification problem.

### **Algorithm steps for CNN:**

1. a) Convolution Operation  
    b) Re LU Layer
2. Pooling
3. Flattening
4. Full Connection

### **1. a) Convolution Operation:**

- The first building block in our plan of attack is convolution operation. In this step, we will touch
- On feature detectors, which basically serve as the neural network's filters.
- We will also discuss feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection and how the findings are mapped out.
- Extract the unique features from the input image. Convolution is a mathematical operation on two functions ( $f$  and  $g$ ) that produces a third function ( $f*g$ ) expressing how the shape of one is modified by the other.

## **b) Re LU Layer:**

- Re LU (Rectified Linear Unit) *Layer*.
- Re LU refers to the Rectifier Unit, the most deployed activation function for the outputs of the CNN neurons.

## **2. Pooling:**

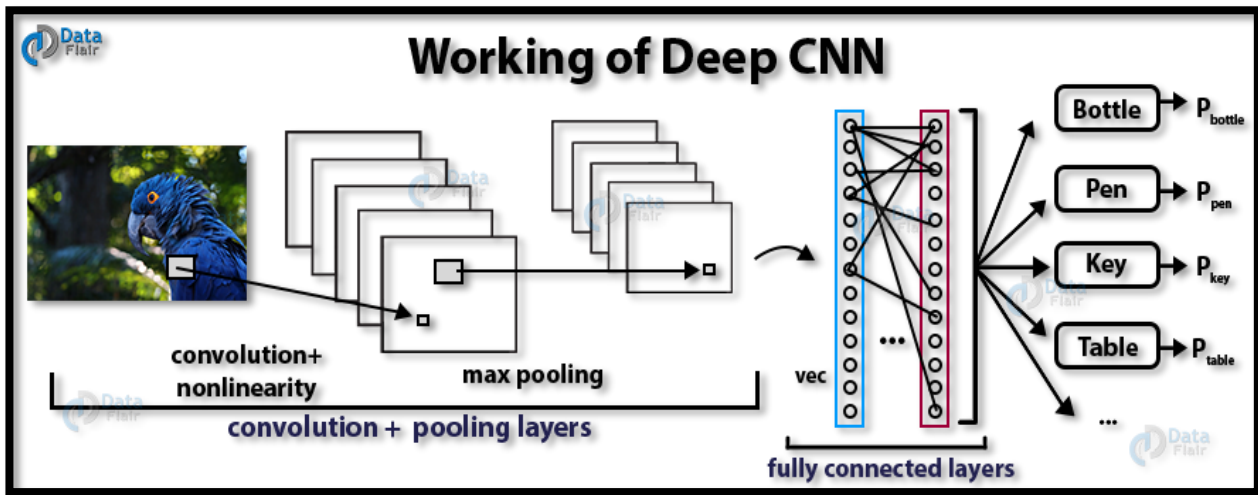
- Pooling is a down-sampling operation that reduces the dimensionality of the feature map. The rectified feature map now goes through a pooling layer to generate a pooled feature map.
- The pooling layer uses various filters to identify different parts of the image like edges, corners, body, feathers, eyes, and beak

## **3. Flattening:**

- Flattening is converting the data into a 1-dimensional array for inputting it to the next layer.
- We flatten the output of the convolution layers to create a single long feature vector. And is connected to the final classification model. The flattened matrix is fed as input to connected layer to classify the image.

## **4. Full Connection:**

- Fully Connected Layers from the last few layers in the network.
- The input to the fully connected layer is the output from the final pooling or Convolution layer.



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.

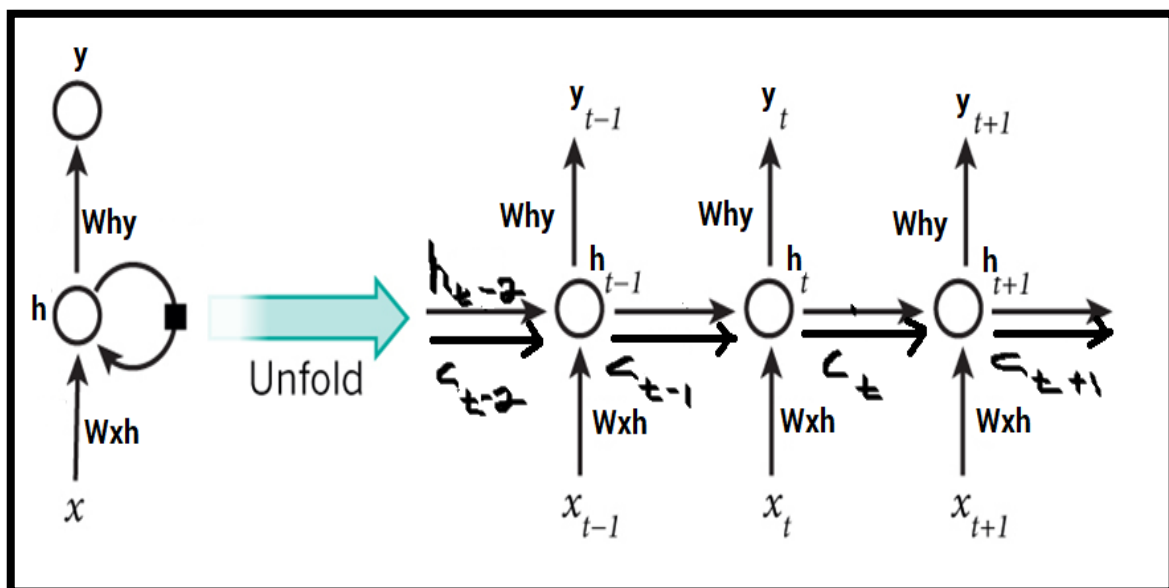
LSTM networks are an extension of recurrent neural networks (RNNs) mainly introduced to handle situations where RNNs fail. Talking about RNN, it is a network that works on the present input by taking into consideration the previous output (feedback) and storing in its memory for a short period of time (short-term memory). Out of its various applications, the most popular ones are in the fields of speech processing, non-Markovian control, and music composition. Nevertheless, there are drawbacks to RNNs. First, it fails to store information for a longer period of time. At times, a reference to certain information stored quite a long time ago is required to predict the current output. But RNNs are absolutely incapable of handling such “long-term dependencies”. Second, there is no finer control over which part of the context needs to be carried forward and how much of the past needs to be ‘forgotten’. Other issues with RNNs are exploding and vanishing gradients (explained later) which occur during the training process of a network through backtracking. Thus, Long Short-Term Memory (LSTM) was brought into the picture. It has been so designed that the vanishing gradient problem is almost completely removed, while the training model is left unaltered. Long time lags in certain problems are bridged using LSTMs where they also handle noise, distributed representations, and continuous values. With LSTMs, there is no need to keep a finite number of states from beforehand as required in the hidden Markov model (HMM). LSTMs provide us with a large range of parameters such as learning rates, and input and output biases. Hence, no need for fine adjustments. The complexity to

update each weight is reduced to  $O(1)$  with LSTMs, similar to that of Back Propagation Through Time (BPTT), which is an advantage.

## ARCHITECTURE:

The basic difference between the architectures of RNNs and LSTMs is that the hidden layer of LSTM is a gated unit or gated cell. It consists of four layers that interact with one another in a way to produce the output of that cell along with the cell state. These two things are then passed onto the next hidden layer. Unlike RNNs which have got the only single neural net layer of tanh, LSTMs comprises of three logistic sigmoid gates and one tanh layer. Gates have been introduced in order to limit the information that is passed through the cell. They determine which part of the information will be needed by the next cell and which part is to be discarded. The output is usually in the range of 0-1 where '0' means 'reject all' and '1' means 'include all'.

## HIDDEN LAYERS OF LSTM:



## APPLICATIONS:

LSTM models need to be trained with a training dataset prior to its employment in real-world applications. Some of the most demanding applications are discussed below:

- Language modelling or text generation, that involves the computation of words when a sequence of words is fed as input. Language models can be operated at the character level, n-gram level, sentence level or even paragraph level.
- Image processing, that involves performing analysis of a picture and concluding its result into a sentence. For this, it's required to have a dataset comprising of a good number of pictures with their corresponding descriptive captions. A model that has already been trained is used to predict features of images present in the dataset.
- Speech and Handwriting Recognition
- Music generation which is quite similar to that of text generation where LSTMs predict musical notes instead of text by analysing a combination of given notes fed as input.
- Language Translation involves mapping a sequence in one language to a sequence in another language. Similar to image processing, a dataset, containing phrases and their translations, is first cleaned and only a part of it is used to train the model. An encoder-decoder LSTM model is used which first converts input sequence to its vector representation (encoding) and then outputs it to its translated version.

### **1.3 PURPOSE:**

Image caption Generator is a popular research area of Artificial Intelligence that deals with image understanding and a language description for that image. Generating well-formed sentences requires both syntactic and semantic understanding of the language. Being able to describe the content of an image using accurately formed sentences is a very challenging task, but it could also have a great impact, by helping visually impaired people better understand the content of images.

This task is significantly harder in comparison to the image classification or object recognition tasks that have been well researched.

The biggest challenge is most definitely being able to create a description that

must capture not only the objects contained in an image, but also express how these objects relate to each other.

## **1.4 SCOPE**

There are many methods in image caption generator and classification processes, but still, this research field is lacking. In addition, there are still no commercial solutions on the market.

In this project, a new approach of using deep learning method was explored to automatically classify and predict the situation in the image. This application will serve as an aid to blind people, students and many more to enabling fast and efficient recognition of image and facilitating the decision-making process easily.

The main goal for the future work will be developing a complete system consisting of server-side components containing a trained model and an application for smart mobile devices with features such as displaying recognized images and suggesting the natural sentences happening in the image. By extending this project, we can achieve a valuable impact on the time management and with efficient results we can make any decision correctly.

## **2. SRS DOCUMENT**

A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform.

### **2.1 FUNCTIONAL REQUIREMENTS**

A Functional Requirement (FR) is a description of the service that the software must offer. It describes a software system or its component. A function is nothing but inputs to the software system, its behavior, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. Functional Requirements are also called Functional Specification.

- Analyzing the situation happening in the image with the algorithm.
- Image based model — Extracts the features of our image.
- Language based model — which translates the features and objects extracted by our image-based model to a natural sentence.

## 2.2 NON-FUNCTIONAL REQUIREMENTS

NON-FUNCTIONAL REQUIREMENT (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability. Non-functional requirements are called qualities of a system, there are as follows:

- **Performance**-The average response time of the system is less.
- **Reliability** - The system is highly reliable.
- **Operability** - The interface of the system will be consistent.
- **Efficiency** - Once user has learned about the system through his interaction, he can perform the task easily.
- **Understandability**-Because of user friendly interfaces, it is more understandable to the users.

## 2.3 MINIMUM HARDWARE REQUIREMENTS

- **Processor** -Intel Core i5
- **Hard Disk** – 1 TB
- **RAM** - 8GB

## 2.4 MINIMUM SOFTWARE REQUIREMENTS

Python based Computer Vision and Deep Learning libraries will be exploited for the development and experimentation of the project.

- **Programming Language** – PYTHON 3.0
- **Operating System** - Windows 10(64 bit)
- **IDE** - Anaconda Navigator

- **Packages** – TensorFlow, Keras, Pillow, Matplotlib, NumPy, Flask.

### **3. ANALYSIS**

#### **3.1 EXISTING SYSTEMS**

So far, the principal method used has been naked eye observation. Individual inspectors' grading findings are frequently inconsistent due to factors such as differing working conditions, personal judgement, and level of weariness. The caption bot, which generates image captioning, is a product of Microsoft, but it is now unavailable owing to inconsistencies in the web page. Some techniques have been found to be erroneous, owing to the presence of noise in the input image. The present visual inspection procedures must be automated in order to attain this goal.

#### **DRAWBACKS OF EXISTING ALGORITHMS**

- When seeing the image with the naked eye, it is difficult to appropriately caption the image.
- The web server isn't up to par, frequently returning a 402 error or failing to reply to requests.
- There are a lot of apps that aid in captioning images, but none of them are 100% accurate.

#### **3.2 PROPOSED ALGORITHM**

The proposed model has been developed to address the shortcomings of existing applications. We used Res-Net architecture to create a CNN model and an LSTM (Long Short-Term Memory) model, as well as various pre-processing techniques using this model, we can build accurate captioning of image. Our application can recognize various types of images.



## **ADVANTAGES OF PROPOSED MODEL**

Our project will provide you with more precise results. Whatever the photographs are like, if they are free of noise and disruption, we will get more accurate findings.

- It saves a lot of time.
- It is also simple to use for those with little technical understanding.
- It is reliable

### **3.3 FEASIBILITY STUDY**

A feasibility study is an examination that considers all important aspects of a project, including economic, technical, legal, and scheduling concerns, in order to determine the likelihood of the project's successful completion. A feasibility study is necessary to determine whether or not a proposed project is feasible. A feasibility study is essentially an evaluation of a proposed plan or project's viability.

**The main objectives of feasibility are mentioned below:**

The main goal of the feasibility study activity is to establish whether the product is technically and financially feasible to develop. A feasibility study should give management enough information to make the following decisions:

- Whether or not the project is feasible.
- To estimate the likelihood of your suggested action being successful.
- Whether or not the finished work will be beneficial to the intended audience.
- To characterize the project's nature and complexity.
- What are the options from which a solution will be picked (in the following phases)?
- To determine if the software satisfies the needs of the organization.

There are various types of feasibility that can be determined. They are:

**Operational** - Define the problem's urgency and the acceptability of any solution; this covers both internal and external difficulties, such as manpower issues, labor objections, manager opposition, organizational disputes, and policies; as well as societal acceptability, legal considerations, and government restrictions.

**Technical** - Is it possible given the current state of technology? Is the technology even possible? Is it included within a certain resource?

**Economic** - Is the project feasible given the available resources? Are the gains that the new system will provide worth the costs? What are the actual and intangible savings that will be realized as a result of the system? What is the status of development and operations?

**Schedule** - Constraints on the project schedule and whether they could be reasonably met.

### **3.3.1 Economic Feasibility:**

Cost/benefit analysis is another term for economic analysis. It is the most common way for determining a new system's effectiveness. The procedure in economic analysis is to evaluate the projected advantages and savings from a prospective system and compare them to the costs. Before the project begins, an economic feasibility analysis is conducted to determine the price and any other costs associated with the scheme. This research also boosts project trustworthiness. It also aids decision-makers in determining whether a planned scheme should be implemented later or now, depending on the organization's financial situation.

The suggested scheme's pricing benefits are also investigated during this review procedure. The following tasks are likewise performed by Economic Feasibility.

- Packaged software/software development costs.
- Cost of doing full system study.
- Is the system economically viable?

### **3.3.2 Technical Feasibility:**

Assessing technical feasibility is an important aspect of deciding resources. It takes into account the project's technological requirements. The technical needs are then compared against the organization's technical capacity. If the internal technological competence is sufficient to satisfy the project needs, the systems project is regarded technically feasible. The analyst must determine whether current technological resources can be utilized system analysts' expertise is useful, as they will be able to answer the question of technical feasibility based on their own experience and contacts with vendors.

The following tasks are likewise performed by Technical Feasibility.

- Is the technology feasible given the existing resources?
- Is the technology capable of handling the problem?
- Determines whether the technology in question is stable and well-established.
- Is the software development technology chosen one that has a large number of users who can be consulted when problems arise or upgrades are needed?

### **3.3.3 Operational Feasibility:**

Operational feasibility refers to how successfully a proposed system solves problems and exploits possibilities discovered during scope definition, as well as how well it meets the criteria determined during the requirements analysis phase of system development. The operational feasibility relates to the availability of operational resources required to expand research results beyond the context in which they were generated, and for which all operational requirements are low and readily met. Furthermore, whatever sensible sacrifices farmers make in tailoring the technology to the limited operational resources available to them would be included in the operational feasibility. Operational Feasibility also carries out responsibilities such as

- Is the existing mode of operation sufficient in terms of response time?
- Is the current mode of operation making the best use of resources?
- Determines whether the software development team's proposed solution is acceptable.

- Is there a good way to control the data in this operation?
- Our project runs on a processor, and the system supports the packages we've installed.

### **3.4 COST BENEFIT ANALYSIS**

The financial and the economic questions during the preliminary investigation are verified to estimate the following:

- The cost of the hardware and software for the class of application being considered.
- Benefits in the form of lower costs
- The proposed system will give the minute information, as a result.
- Performance is improved which in turn may be expected to provide increased profits.
- This feasibility checks whether the system can be developed with the available funds.
- This can be done economically if planned judiciously, so it is economically feasible.
- The cost of the project depends upon the number of man-hours required.

## **4. SOFTWARE DESCRIPTION**

### **4.1 Flask**

Flask is an API of Python that allows us to build up web-applications. It was developed by Armin Ronacher. Flask is considered more Pythonic than the Django web framework because in common situations the equivalent Flask web application is more explicit. Flask is also easy to get started with as a beginner because there is little boilerplate code for getting a simple app up and running. Flask (source code) is a Python web framework built with a small core and easy-to-extend philosophy. provides tools, Libraries and technologies that allow you to build a web application.

### **4.2 Keras**

Keras is an open-source neural-network library written in Python. It can run on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research

effort of project ONEIROS (Open ended Neuro- Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet also is the author of the Xception deep neural network model. In 2017, Google's TensorFlow team decided to support Keras in TensorFlow's core library. Chollet explained that Keras was conceived to be an interface rather than a standalone machine learning framework. It offers a higher level, more intuitive set of abstractions that make it easy to develop deep learning models regardless of the computational backend used.

### **4.3 TensorFlow**

TensorFlow is an open-source machine learning framework for all developers. It is used for implementing machine learning and deep learning applications. To develop and research fascinating ideas on artificial intelligence, the Google team created TensorFlow. TensorFlow is designed in Python programming language, hence it is considered an easy to-understand framework. TensorFlow is mainly used for: Classification, Perception, Understanding, Discovering, Prediction and Creation.

### **4.4 Matplotlib**

Matplotlib is a plotting library for python. Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002. Matplotlib comes with a wide variety of plots. Plots help to understand trends, patterns, and to make correlations. It consists of various plots like line, bar, scatter etc. `pyplot ()` is the most important function in matplotlib library, which is used to plot 2D data.

### **4.5 Numpy**

NumPy stands for Numerical Python. NumPy is a python library used for working with arrays. It also has functions for working in the domain of linear algebra, Fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an opensource project, and you can use it Freely. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. It is optimized to work with the latest CPU architectures. NumPy arrays are stored at one

continuous place in memory unlike lists, so processes can access and manipulate them very efficiently.

## **4.6 LSTM**

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can process not only single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition, speech recognition and anomaly detection in network traffic or IDSs (intrusion detection systems).

# **5. PROJECT DESCRIPTION**

## **5.1 PROBLEM DEFINITION**

You saw an image and your brain readily deduced what it represented, but can a computer deduce what it represents? We developed model that can generate captions for images thanks to advances in deep learning techniques, the availability of large datasets, and computer power. This is exactly what we'll achieve in this Python-based project, where we'll mix Convolutional Neural Networks and a type of Recurrent Neural Network (LSTM) deep learning approach.

## **5.2 PROJECT OVERVIEW**

Image caption generator employs computer vision and natural language processing techniques to recognize the context of an image and explain it in a natural language like English. The goal of our project is to study the ideas of a CNN and LSTM model, and then use CNN and LSTM to develop a workable model of an Image caption generator. We will use CNN (Convolutional Neural Networks) and LSTM to construct the caption generator in this Python project (Long short-term memory). The image characteristics will be taken from Resnet50, a CNN model trained on the flickr8k-sau dataset, and then fed into the LSTM model, which will generate the image descriptions.

The steps involved in our project are:

- 1.Collection of datasets.

2. Extracting the features of image using Resnet50 model.
3. Passing the output of the Resnet50 model as input to the developed model and training of the model.
4. To maximize accuracy, repeat steps 2 and 3 for varied ratios of training and testing data.

The output of our project provides a caption for the given image.

## **5.3 MODULE DESCRIPTION**

### **5.3.1 FLASK FRAMEWORK**

Flask is a Python API that allows us to create web-based applications. Armin Ronacher was the one who came up with the idea. Flask is a WSGI web application framework that is lightweight. It's built to make getting started simple and quick, with the flexibility to scale up to more sophisticated projects. Flask's framework is more explicit than Django's, and it's also easier to learn because it requires less basic code to create a simple web application. A Web Application Framework, often known as a Web Framework, is a set of modules and libraries that allow programmers to create applications without having to write low-level code such as protocols, thread management, and so on. The WSGI (Web Server Gateway Interface) toolkit and the Jinja2 template engine are the foundations of Flask.

For the installation of Flask, you'll need Python 2.6 or above. To begin, import Flask from the flask package into any Python IDE. Flask makes recommendations but does not impose any dependencies or project structure. The developer is free to use whatever tools and libraries they wish. The community has created a number of extensions that make adding additional functionality simple.

### **5.3.2 MODEL**

#### **DEEP LEARNING**

Deep learning is a machine learning technique that allows computers to learn by example in the same way that humans do. Deep learning is a critical component of self-driving automobiles, allowing them to detect a stop sign or discriminate

between a pedestrian and a lamppost. It enables voice control in consumer electronics such as phones, tablets, televisions, and hands-free speakers. Deep learning has gotten a lot of press recently, and with good cause. It's accomplishing accomplishments that were previously unattainable.

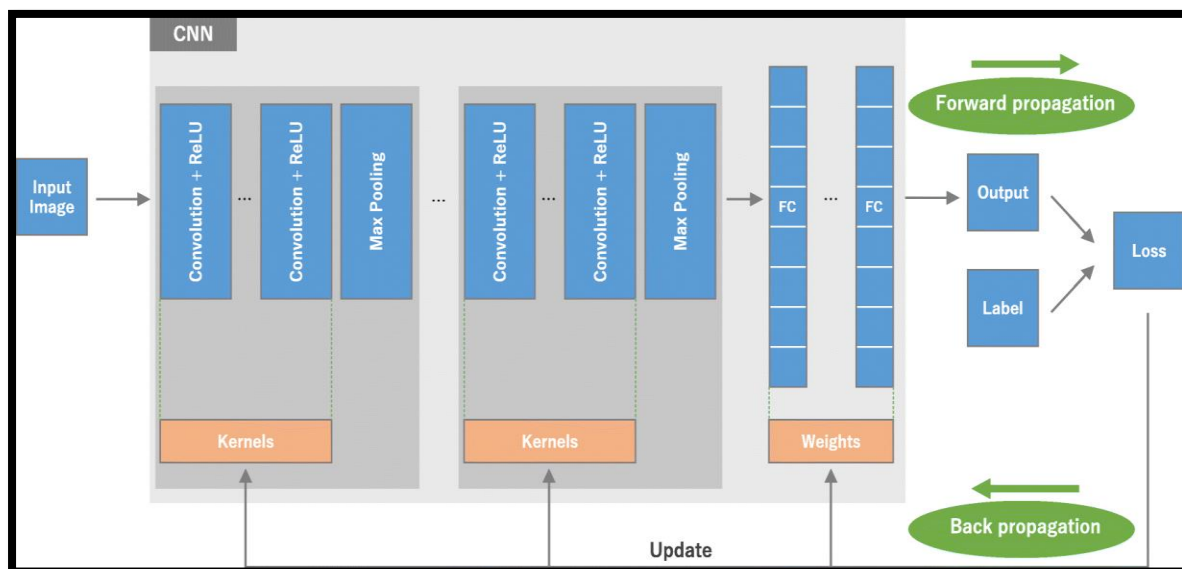
A computer model learns to execute categorization tasks directly from images, text, or sound in deep learning. Deep learning models can attain state-of-the-art accuracy, even surpassing human performance in some cases. Models are trained utilizing a huge quantity of labelled data and multilayer neural network topologies.

### **CNN (Convolutional Neural Network)**

CNN is a deep learning model for processing data with a grid pattern, such as photographs, that is inspired by the organization of animal visual cortex [13, 14] and meant to learn spatial hierarchies of characteristics, from low- to high-level patterns, automatically and adaptively. Convolutional neural networks (CNNs) are made up of three types of layers (or building blocks): convolution, pooling, and fully connected layers. The first two layers, convolution and pooling, extract features, while the third, a fully linked layer, transfers those features into final output, such as classification. A convolution layer is an important part of CNN, which is made up of a stack of mathematical operations like convolution, which is a specific sort of linear operation.

Pixel values in digital images are stored in a two-dimensional (2D) grid, i.e., an array of numbers, and a small grid of parameters called kernel, an optimizable feature extractor, is applied at each image position, making CNNs highly efficient for image processing because a feature can appear anywhere in the image. Extracted features can evolve hierarchically and progressively more complicated as one layer feeds its output into the next layer. Training is the process of adjusting parameters like kernels in order to reduce the disparity between outputs and ground truth labels using optimization algorithms like backpropagation and gradient descent, among others.





## The different layers of a CNN

The convolutional layer, the pooling layer, the ReLU correction layer, and the fully-connected layer are the four types of layers in a convolutional neural network.

## The convolutional layer

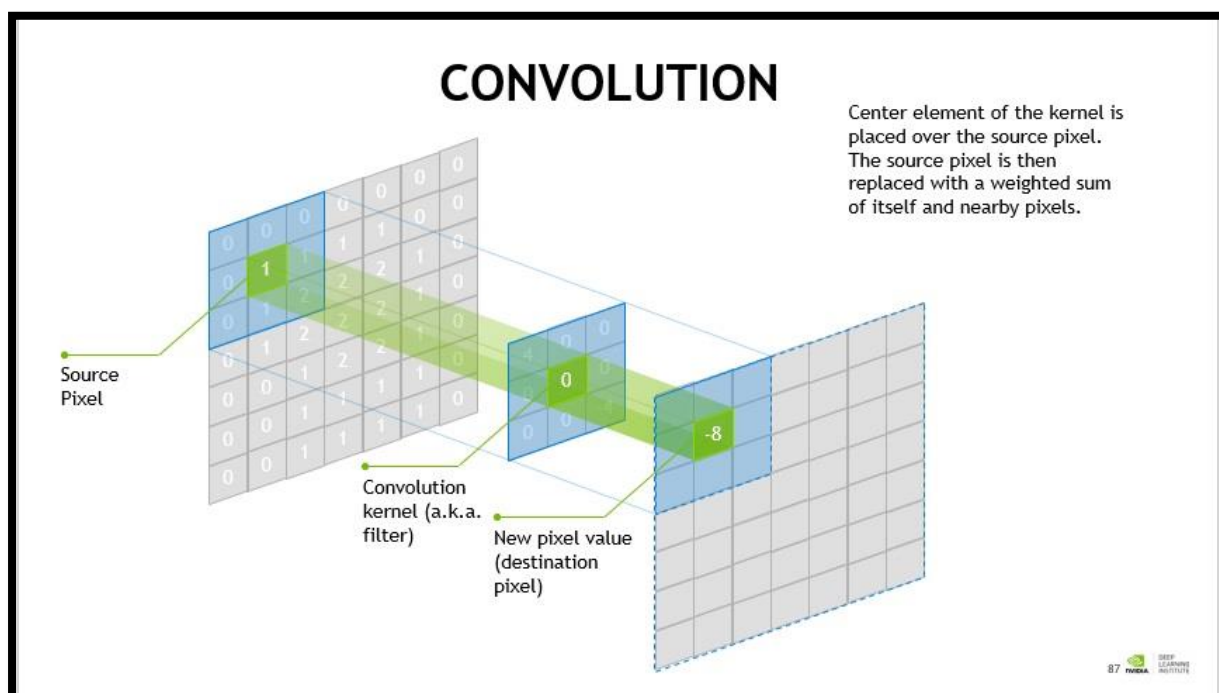
Convolutional neural networks' fundamental component is the convolutional layer, which is always at least the first layer.

Its goal is to find a set of features in the photos that are given to it as input. This is accomplished by convolution filtering, which involves "dragging" a window that represents a feature on the picture and calculating the convolution product between the feature and each area of the scanned image. In this case, a feature is viewed as a filter, and the two terms are interchangeable.

As a result, the convolutional layer takes multiple images as input and calculates the convolution of each with each filter. The filters match the features we're looking for in the photographs to a tree.

We get a feature map for each pair (image, filter) that shows us where the features are in the image: the greater the value, the more the corresponding location in the image resembles the feature.

Unlike previous approaches, features are learned by the network during the training phase rather than being pre-defined according to a particular formalism (for example, SIFT). The convolution layer weights are referred to as filter kernels. They are started and then updated via gradient descent backpropagation.



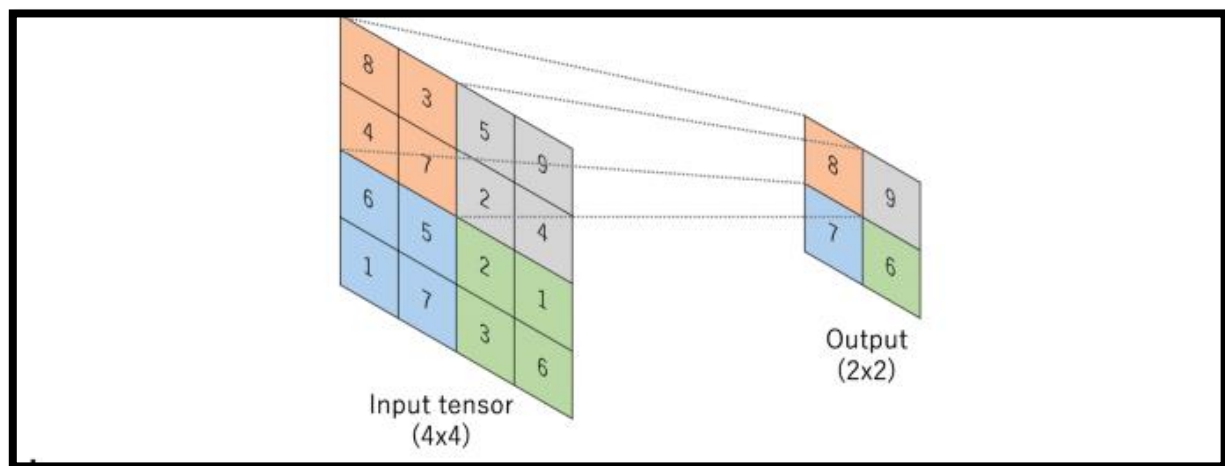
## Convolution Layer

### Pooling layer

A pooling layer performs a standard down sampling operation on the feature maps, reducing their in-plane dimensionality in order to introduce translation invariance for tiny shifts and distortions and reducing the number of learnable parameters. It's worth noting that none of the pooling layers have learnable parameters, although filter size, stride, and padding are hyperparameters in pooling operations, much like they are in convolution operations.

### Max pooling

The most popular form of pooling operation is max pooling, which extracts patches from the input feature maps, outputs the maximum value in each patch, and discards all the other values. A max pooling with a filter of size  $2 \times 2$  with a stride of 2 is commonly used in practice. This down samples the in-plane dimension of feature maps by a factor of 2. Unlike height and width, the depth dimension of feature maps remains unchanged.



## GLOBAL AVERAGE POOLING

A global average pooling operation is also worth mentioning [20]. A global average pooling is an extreme type of down sampling in which a feature map with a height and width of height and width is down sampled into a 1 1 array by simply taking the average of all the elements in each feature map, but the depth of feature maps is preserved. Before the fully connected layers, this step is usually performed only once. The following are some of the benefits of using global average pooling: (1) decreases the number of parameters that can be learned, and (2) allows the CNN to accept inputs of varying sizes.

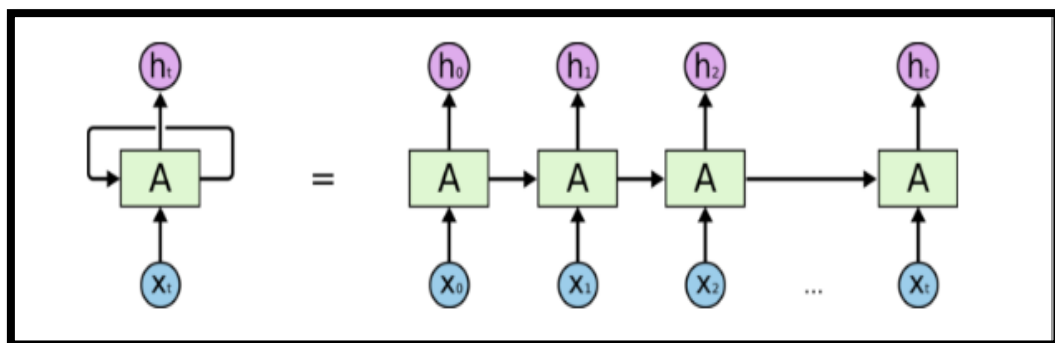
## FULLY CONNECTED LAYER

The final convolution or pooling layer's output feature maps are typically flattened, that is, converted into a one-dimensional (1D) array of numbers (or vector), and connected to one or more fully connected layers, also known as dense layers, in which each input is connected to each output by a learnable weight. Once the features extracted by the convolution layers and down sampled by the pooling layers are formed, they are transferred to the network's final outputs, such as the probabilities for each class in classification tasks, by a subset of fully connected layers. The number of output nodes in the final fully connected layer is usually equal to the number of classes. Following each fully connected layer is a nonlinear function, such as ReLU, as described above.

## Recurrent Neural Network (RNN)

A feedforward neural network with an internal memory is known as a recurrent neural network. RNN is recurrent in nature since it executes the same function for each data input, and the current input's outcome is dependent on the previous computation. The output is replicated and transmitted back into the recurrent network when it is created. It evaluates the current input as well as the output it has learned from the prior input when making a decision.

RNNs, unlike feedforward neural networks, may process sequences of inputs using their internal state (memory). As a result, activities like unsegmented, connected handwriting recognition or speech recognition are possible. All of the inputs in other neural networks are independent of one another. In an RNN, however, all of the inputs are connected to one another.



**An unrolled recurrent neural network**

It first extracts  $X(0)$  from the series of inputs, then outputs  $h(0)$ , which, along with  $X(1)$ , serves as the input for the next phase. As a result, the next step's inputs are  $h(0)$  and  $X(1)$ . Similarly, the following step's input is  $h(1)$ , and the next step's input is  $X(2)$ , and so on. As a result, it remembers the context while training.

The formula for the current state is

$$h_t = f(h_{t-1}, X_t)$$

Applying Activation Function:

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}X_t)$$

W is weight, h is the single hidden vector,  $W_{hh}$  is the weight at previous hidden state,  $W_{xh}$  is the weight at current input state, tanh is the Activation function, that implements a Non-linearity that squashes the activations to the range[-1.1].

Output:

$$Y_t = W_{hy}h_t$$

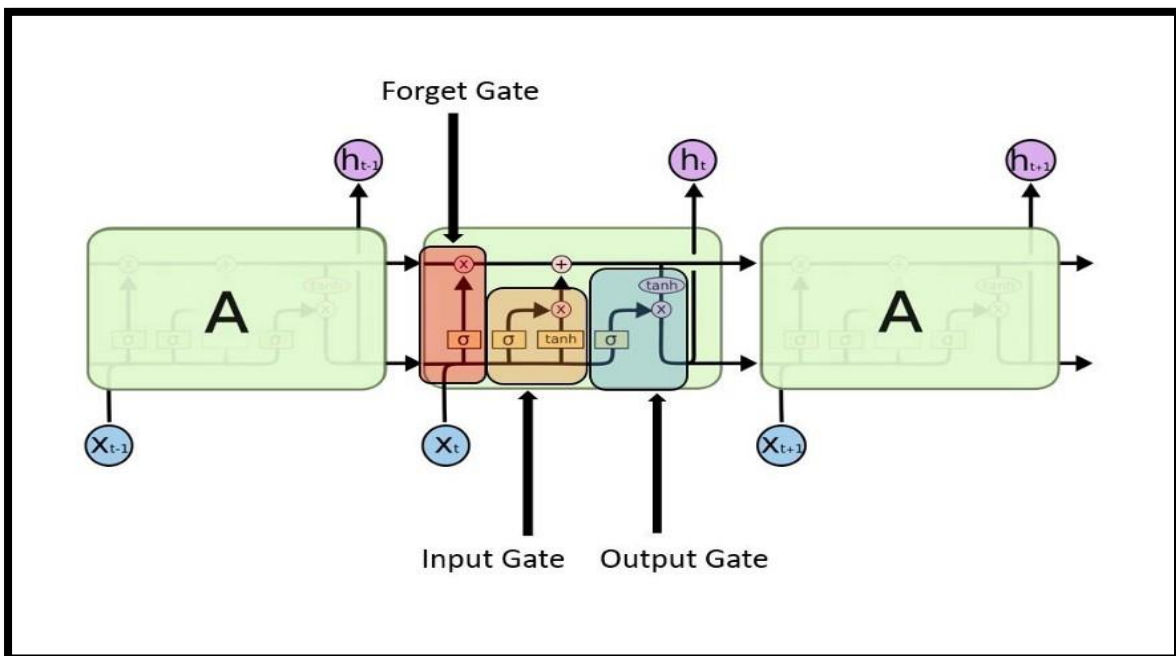
$Y_t$  is the output state.  $W_{hy}$  is the weight at the output state.

## Disadvantages of RNN

1. Gradient vanishing and exploding problems.
2. Training an RNN is a very difficult task.
3. It cannot process very long sequences if using tanh *or* relu as an activation function.

## Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a modified version of recurrent neural networks that make it simpler to recall information from the past. Here, the RNN's vanishing gradient problem is solved. Given time lags of uncertain duration, LSTM is well-suited to identify, analyse, and predict time series. Back-propagation is used to train the model. Three gates are present in an LSTM network:



1. **Input gate** — figure out which input value should be utilised to change the memory. The sigmoid function determines which numbers are allowed to pass through 0,1. and the tanh function assigns weight to the data supplied, determining their importance on a scale of -1 to 1
2. **Forget gate** — discover what details to be discarded from the block. It is decided by the **sigmoid function**. it looks at the previous state( $h_{t-1}$ ) and the content input ( $x_t$ ) and outputs a number between 0(omit this) and 1(keep this) for each number in the cell state  $C_{t-1}$ .

## ARCHITECTURE

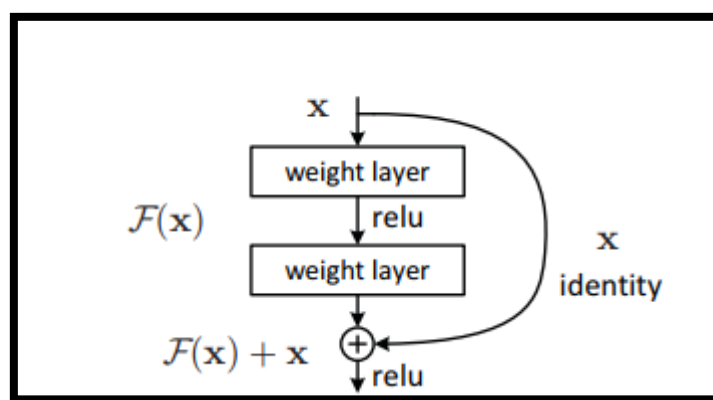
Every consecutive winning design employs more layers in a deep neural network to minimise the error rate after the first CNN-based architecture (AlexNet) won the ImageNet 2012 competition. This works for a small number of layers, but as the number of layers grows, a typical problem in deep learning called Vanishing/Exploding gradient emerges. As a result, the gradient becomes 0 or too huge. As a result, as the number of layers increases, so does the training and test error rate.

## ResNet:

ResNet is a well-known deep learning model that was first introduced in a paper by Shaoqing Ren, Kaiming He, Jian Sun, and Xiangyu Zhang. In 2015, the work was titled "Deep Residual Learning for Image Recognition." ResNet is one of the most widely used and successful deep learning models to date.

### Residual Blocks:

The introduction of these Residual blocks alleviated the challenge of training very deep networks, and the ResNet model is built up of these blocks.



The introduction of these Residual blocks alleviated the challenge of training very deep networks, and the ResNet model is built up of these blocks.

The first thing we note in the above diagram is that there is a direct connection that skips several of the model's levels. The 'skip connection,' as it is known, is at the heart of residual blocks. Because of the skip connection, the output is not the same. Without the skip connection, input 'X' is multiplied by the layer's weights, then a bias term is added.

The activation function,  $f()$ , is then applied, and the result is  $H(x)$ .

$$H(x)=f(wx + b) \text{ or } H(x)=f(wx + b) \text{ or } H(x)=f(wx + b) \text{ or } H(x)= (x)$$

The output of  $H(x)$  has been modified due to the introduction of a new skip connection mechanism.

$$H(x)=f(x)+x$$

However, when using a convolutional layer or pooling layer, the input dimension may differ from the output dimension. As a result, the following two ways can be used to solve this issue:

To expand the size of Zero, it is padded with the skip connection.

To make the input fit the dimensions, 11 convolutional layers are added. The result is as follows:

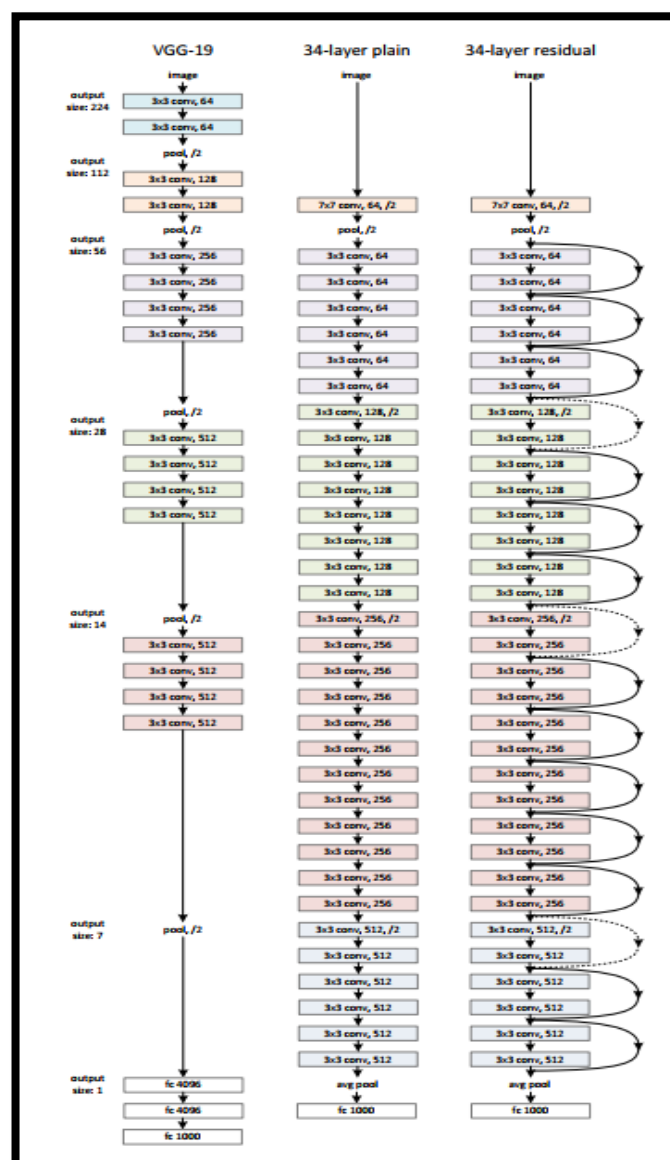
$$H(x)=f(x)+w1.x$$

When utilising the first strategy, no additional parameter  $w1$  is added.

ResNet uses a method called skip connections.

## Architecture of ResNet

The architecture is inspired on VGG-19 and has a 34-layer plain network to which shortcut and skip connections are added. As seen in the diagram below, these skip connections or residual blocks change the architecture into a residual network.





## 6. DEVELOPMENT

### 6.1. DATASET USED

A DATASET is a collection or set of data. This information is usually presented in a tabular format. Each column denotes a distinct variable. According to the question, each row belongs to a certain member of the data set. This is part of the DATA MANAGEMENT process.

We have collected dataset of images and their respective captions from KAGGEL.COM.flicker8k\_sau dataset contains 8000 images and 5 captions to each image. It is small in size, as a result, the model may be quickly trained on low-end laptops and PCs.

### 6.2 Sample Code:

```
#Importing numpy,pandas,opencv
```

```
import numpy as np
```

```
import pandas as pd
```

```
import cv2
```

```
import os
```

```
from glob import glob
```

```
#getting the path of the images in the dataset
```

```
images_path = '../input/flicker8k-sau/Flickr_Data/Images/'
```

```
images = glob(images_path+'*.jpg')
```

```
len(images)
```

```
#Importing matplotlib-for displaying the images
```

```
import matplotlib.pyplot as plt
```

### **#Displaying images using matplotlib**

```
for i in range(5):  
  
    plt.figure()  
  
    img = cv2.imread(images[i])  
  
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)  
  
    plt.imshow(img)
```

### **#Importing ResNet50 model from keras**

```
from keras.applications import ResNet50  
  
incept_model = ResNet50(include_top=True)
```

### **#Removing the last output layer(predictions(Dense Layer))**

```
from keras.models import Model  
  
last = incept_model.layers[-2].output  
  
modele = Model(inputs = incept_model.input, outputs = last)  
  
modele.summary()
```

### **#features of image are extracted in the form of vector of size (2048,) as output from ResNet Model**

```
images_features = {}  
  
count = 0  
  
for i in images:  
  
    img = cv2.imread(i)  
  
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

```

img = cv2.resize(img, (224,224))

img = img.reshape(1,224,224,3)

pred = modele.predict(img).reshape(2048,)

img_name = i.split('/')[-1]

images_features[img_name] = pred

count += 1

if count > 1499:

    break

elif count % 50 == 0:

    print(count)

#Getting the path of the captions

caption_path = '../input/flickr8k-
sau/Flickr_Data/Flickr_TextData/Flickr8k.token.txt'

#Reading the caption_path

captions = open(caption_path, 'rb').read().decode('utf-8').split('\n')

```

**#Froming key-value pairs where key is the name of the image and values are the 5 types of captions to the image**

```
captions_dict = {}

for i in captions:

    try:

        img_name = i.split('\t')[0][:2]

        caption = i.split('\t')[1]

        if img_name in images_features:

            if img_name not in captions_dict:

                captions_dict[img_name] = [caption]

            else:

                captions_dict[img_name].append(caption)

        except:

            pass
```

**#Functions return the test with startofseq as starting string and endofseq as the ending string**

```
def preprocessed(txt):

    modified = txt.lower()

    modified = 'startofseq ' + modified + ' endofseq'
```

```
return modified
```

### **#Forming sentences with startofseq+caption+endofseq**

```
for k,v in captions_dict.items():
```

```
    for vv in v:
```

```
        captions_dict[k][v.index(vv)] = preprocessed(vv)
```

### **#Counting the words and storing in a dictionary named count\_words**

```
count_words = {}
```

```
for k,vv in captions_dict.items():
```

```
    for v in vv:
```

```
        for word in v.split():
```

```
            if word not in count_words:
```

```
                count_words[word] = 0
```

```
            else:
```

```
                count_words[word] += 1
```

### **#Converting the words into numbers as the model deals with numbers**

```
for k, vv in captions_dict.items():
```

```
    for v in vv:
```

```
        encoded = []
```

```

for word in v.split():

    if word not in new_dict:

        encoded.append(new_dict['<OUT>'])

    else:

        encoded.append(new_dict[word])

captions_dict[k][vv.index(v)] = encoded

#importing to_categorical,pad_sequences

from keras.utils import to_categorical

from keras.preprocessing.sequence import pad_sequences

#Finding the max_length among all the captions

MAX_LEN = 0

for k, vv in captions_dict.items():

    for v in vv:

        if len(v) > MAX_LEN:

            MAX_LEN = len(v)

            print(v)

#Preparing Training data

Batch_size = 5000

VOCAB_SIZE = len(new_dict)

```

```

def generator(photo, caption):

    n_samples = 0

    X = []

    y_in = []

    y_out = []

    for k, vv in caption.items():

        for v in vv:

            for i in range(1, len(v)):

                X.append(photo[k])

                in_seq= [v[:i]]

                out_seq = v[i]

                in_seq = pad_sequences(in_seq, maxlen=MAX_LEN, padding='post',
truncating='post')[0]

                out_seq = to_categorical([out_seq], num_classes=VOCAB_SIZE)[0]

                y_in.append(in_seq)

                y_out.append(out_seq)

```

```
return X, y_in, y_out
```

### **#Generating the training data**

```
X, y_in, y_out = generator(images_features, captions_dict)
```

```
#Changing the data into numpy array
```

```
X = np.array(X)
```

```
y_in = np.array(y_in, dtype='float64')
```

```
y_out = np.array(y_out, dtype='float64')
```

### **#Importing all the layers required to build the model**

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

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

```
from keras.utils import plot_model
```

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

```
from keras.layers import Input
```

```
from keras.layers import Dense
```

```
from keras.layers import LSTM
```

```
from keras.layers import Embedding
```

```
from keras.layers import Dropout
```

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



```
from keras.callbacks import ModelCheckpoint

from keras.layers import Dense, Flatten, Input, Convolution2D, Dropout, LSTM,
TimeDistributed, Embedding, Bidirectional, Activation,
RepeatVector, Concatenate

from keras.models import Sequential, Model

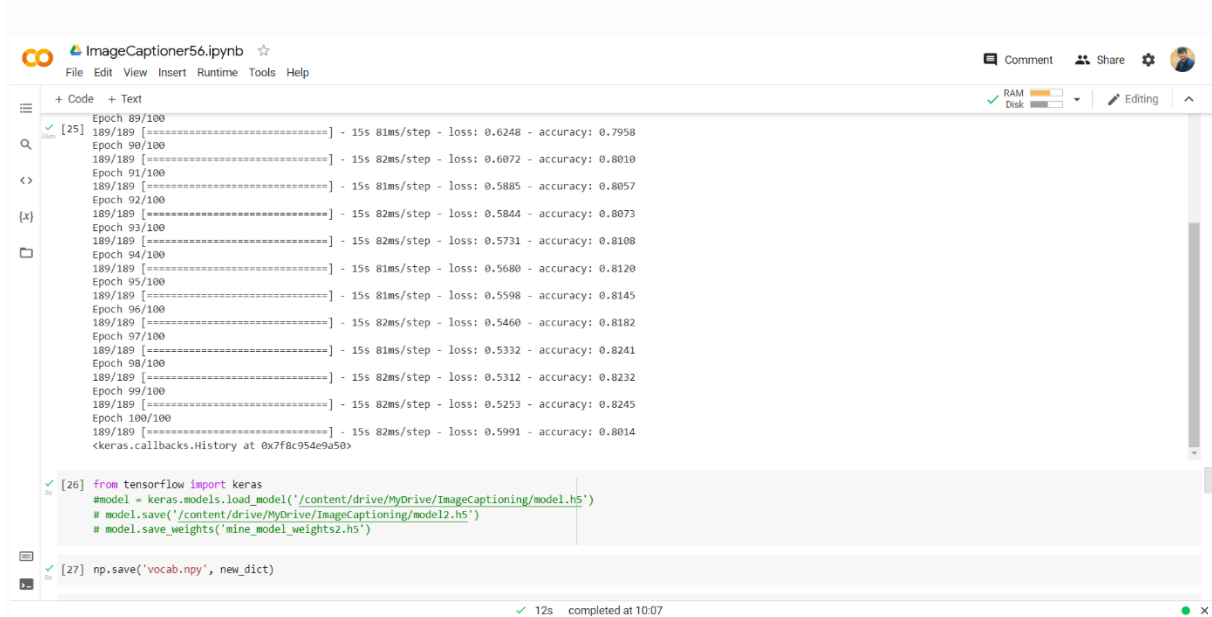
#Training of the model

model.fit([X, y_in], y_out, batch_size=512, epochs=50)

#Inversing the dictionary

inv_dict = {v:k for k, v in new_dict.items()}
```

## 6.3 RESULTS



```
Epoch 89/100
189/189 [=====] - 15s 81ms/step - loss: 0.6248 - accuracy: 0.7958
Epoch 90/100
189/189 [=====] - 15s 82ms/step - loss: 0.6072 - accuracy: 0.8010
Epoch 91/100
189/189 [=====] - 15s 81ms/step - loss: 0.5885 - accuracy: 0.8057
Epoch 92/100
189/189 [=====] - 15s 82ms/step - loss: 0.5844 - accuracy: 0.8073
Epoch 93/100
189/189 [=====] - 15s 82ms/step - loss: 0.5731 - accuracy: 0.8108
Epoch 94/100
189/189 [=====] - 15s 81ms/step - loss: 0.5680 - accuracy: 0.8120
Epoch 95/100
189/189 [=====] - 15s 81ms/step - loss: 0.5598 - accuracy: 0.8145
Epoch 96/100
189/189 [=====] - 15s 82ms/step - loss: 0.5460 - accuracy: 0.8182
Epoch 97/100
189/189 [=====] - 15s 81ms/step - loss: 0.5332 - accuracy: 0.8241
Epoch 98/100
189/189 [=====] - 15s 82ms/step - loss: 0.5312 - accuracy: 0.8232
Epoch 99/100
189/189 [=====] - 15s 82ms/step - loss: 0.5253 - accuracy: 0.8245
Epoch 100/100
189/189 [=====] - 15s 82ms/step - loss: 0.5091 - accuracy: 0.8014
<keras.callbacks.History at 0x7f8c954e9a50>

[26] from tensorflow import keras
      #model = keras.models.load_model('/content/drive/MyDrive/ImageCaptioning/model.h5')
      # model.save('/content/drive/MyDrive/ImageCaptioning/model2.h5')
      # model.save_weights('mine_model_weights2.h5')

[27] np.save('vocab.npy', new_dict)
```

**Accuracy Score:80%**

## 7.0 TESTING

### 7.1 INTRODUCTION OF TESTING

Software testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves the execution of a software component or system component to evaluate one or more properties of interest. It is required for evaluating the system. This phase is the critical phase of software quality assurance and presents the ultimate view of coding.

### Importance of Testing

The importance of software testing is imperative. A lot of times this process is skipped, therefore, the product and business might suffer. To understand the importance of testing, here are some key points to explain

- Software Testing saves money
- Provides Security
- Improves Product Quality
- Customer satisfaction

Testing is of different ways The main idea behind the testing is to reduce the errors and do it with a minimum time and effort.

### **Benefits of Testing**

- **Cost-Effective:** It is one of the important advantages of software testing. Testing any IT project on time helps you to save your money for the long term. In case if the bugs caught in the earlier stage of software testing, it costs less to fix.
- **Security:** It is the most vulnerable and sensitive benefit of software testing. People are looking for trusted products. It helps in removing risks and problems earlier.
- **Product quality:** It is an essential requirement of any software product. Testing ensures a quality product is delivered to customers.
- **Customer Satisfaction:** The main aim of any product is to give satisfaction to their customers. UI/UX Testing ensures the best user experience.

### **Different types of Testing**

**Unit Testing:** Unit tests are very low level, close to the source of your application. They consist in testing individual methods and functions of the classes, components or modules used by your software. Unit tests are in general quite cheap to automate and can be run very quickly by a Continuous integration server.

**Integration Testing:** Integration tests verify that different modules or services used by your application work well together. For example, it can be testing the interaction with the database or making sure that microservices work together as expected. These types of tests are more expensive to run as they require multiple parts of the application to be up and running.

**Functional Tests:** Functional tests focus on the business requirements of an application. They only verify the output of an action and do not check the intermediate states of the system when performing that action. There is

sometimes a confusion between integration tests and functional tests as they both require multiple components to interact with each other. The difference is that an integration test may simply verify that you can query the database while a functional test would expect to get a specific value from the database as defined by the product requirements.

**Regression Testing:** Regression testing is a crucial stage for the product & very useful for the developers to identify the stability of the product with the changing requirements. Regression testing is a testing that is done to verify that a code change in the software does not impact the existing functionality of the product.

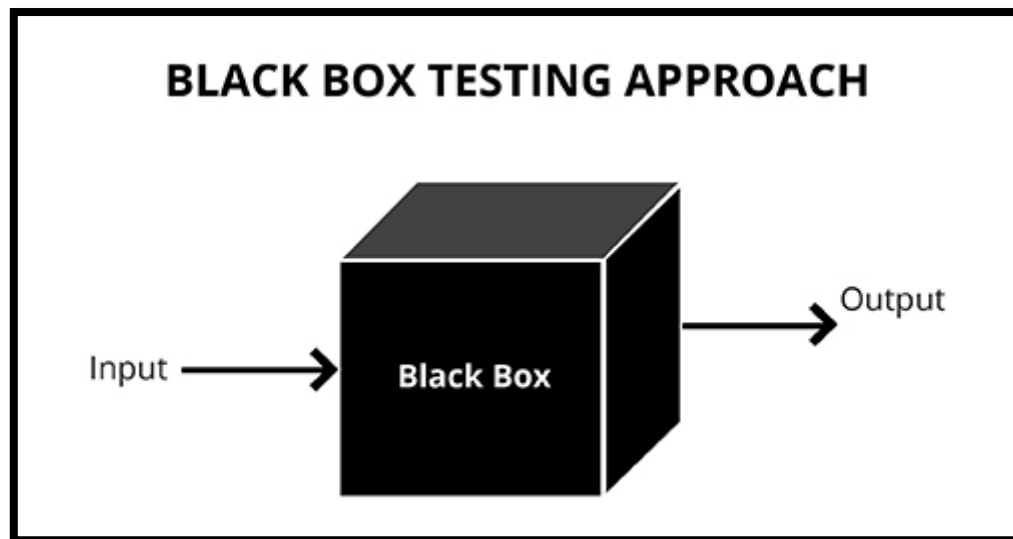
**System Testing:** System testing of software or hardware is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements. System testing is a series of different tests whose primary purpose is to fully exercise the computer-based system.

**Performance Testing:** It checks the speed, response time, reliability, resource usage, scalability of a software program under their expected workload. The purpose of Performance Testing is not to find functional defects but to eliminate performance bottlenecks in the software or device.

**Alpha Testing:** This is a form of internal acceptance testing performed mainly by the in-house software QA and testing teams. Alpha testing is the last testing done by the test teams at the development site after the acceptance testing and before releasing the software for the beta test. It can also be done by the potential users or customers of the application. But still, this is a form of in-house acceptance testing.

**Beta Testing:** This is a testing stage followed by the internal full alpha test cycle. This is the final testing phase where the companies release the software to a few external user groups outside the company test teams or employees. This initial software version is known as the beta version. Most companies gather user feedback in this release.

**Black Box Testing:** It is also known as Behavioral Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.

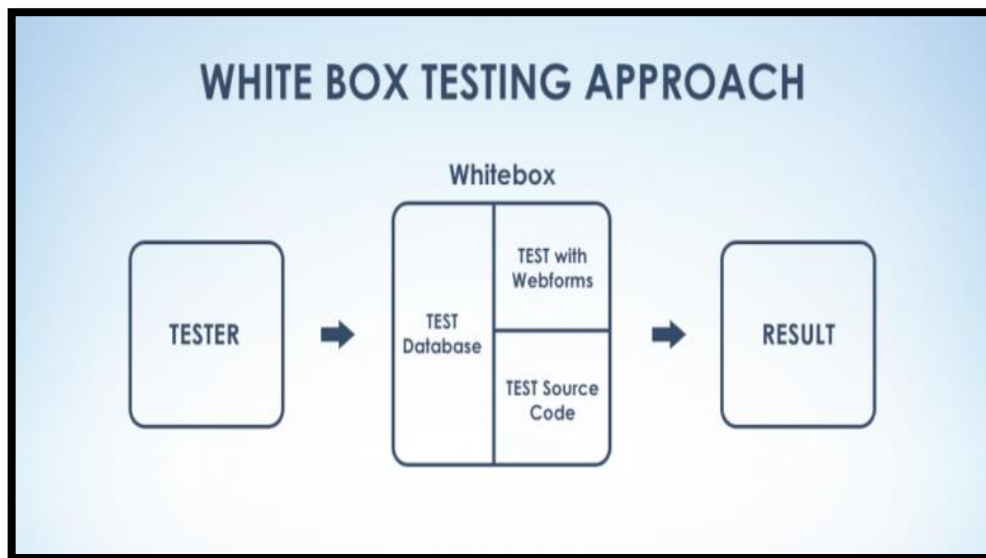


This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. This method attempts to find errors in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structures or external database access
- Behavior or performance errors
- Initialization and termination errors

**White Box Testing:** White box testing (also known as Clear Box Testing, Open Box Testing, Glass Box Testing, Transparent Box Testing, Code-Based Testing or Structural Testing) is a software testing method in which the internal structure/design/implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming know-how and the implementation knowledge is essential. White box testing is testing beyond the user interface and into the nitty-gritty of a system. This method is named so

because the software program, in the eyes of the tester, is like a white/transparent box; inside which one clearly sees.



## 7.2 TEST CODE

#Converting the image into input format

```
def getImage(x):
```

```
    test_img_path = images[x]
```

```
    test_img = cv2.imread(test_img_path)
```

```
    test_img = cv2.cvtColor(test_img, cv2.COLOR_BGR2RGB)
```

```

test_img = cv2.resize(test_img, (224,224))

test_img = np.reshape(test_img, (1,224,224,3))

return test_img

#Testing (Predicting caption to an image.
for i in range(5):

    no = np.random.randint(3000,7000,(1,1))[0,0]
    test_feature = modele.predict(getImage(no)).reshape(1,2048)

    test_img_path = images[no]
    test_img = cv2.imread(test_img_path)
    test_img = cv2.cvtColor(test_img, cv2.COLOR_BGR2RGB)

    text_inp = ['startofseq']

    count = 0
    caption = ""
    while count < 25:

```

```

count += 1

encoded = []

for i in text_inp:

    encoded.append(new_dict[i])

encoded = [encoded]

encoded = pad_sequences(encoded, padding='post', truncating='post',
maxlen=MAX_LEN)

prediction = np.argmax(model.predict([test_feature, encoded]))

sampled_word = inv_dict[prediction]

caption = caption + ' ' + sampled_word

if sampled_word == 'endofseq':

    break

```



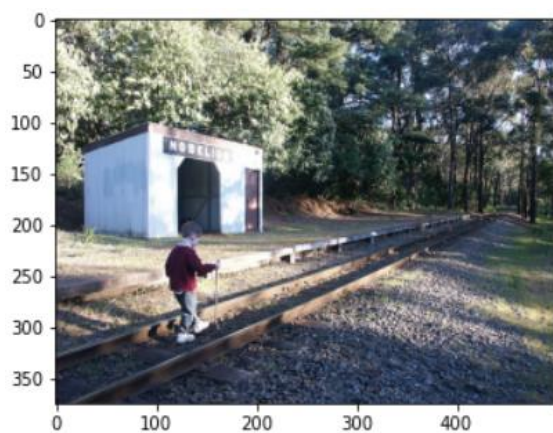
```
text_inp.append(sampled_word)
```

```
plt.figure()
```

```
plt.imshow(test_img)
```

```
plt.xlabel(caption)
```

### 7.3 TEST CASE



a person is riding a rope that has a brown dog. endofseq.



a black dog climbing out of the pool. endofseq



a little girl sitting in a colorful toy car. endofseq

## REFERENCE LINKS

- <https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9>
- <https://towardsdatascience.com/understand-the-architecture-of-cnn-90a25e244c7#:~:text=The%20different%20layers%20of%20a,and%20the%20fully%2Dconnected%20layer.>
- <https://www.analyticsvidhya.com/blog/2021/06/build-resnet-from-scratchwithpython/#:~:text=ResNet%20architecture%20uses%20the%20CNN,batchnorm2d%20after%20each%20conv%20layer.&text=Then%20create%20a%20ResNet%20class,and%20the%20number%20of%20classes>
- <https://aditi-mittal.medium.com/understanding-rnn-and-lstm-f7cdf6dfc14e>
- [https://www.ripublication.com/ijaer18/ijaerv13n9\\_102.pdf](https://www.ripublication.com/ijaer18/ijaerv13n9_102.pdf)