Dissertation on

"Image Captioning Web app using Flask and Machine Learning"

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology
in
Computer Science & Engineering

UE18CS390B – Capstone Project Phase - 2

Submitted by:

RAJ.NAGRANI	**PES2201800177**
MANIKYA.PRAKASH.SARASHETTY	**PES2201800301**
NANDINI.R.SONTH	**PES2201800401**
RAMYASHREE.J.R	**PES2201800728**

Under the guidance of

Prof. Ruhi Dubey
Assistant Professor
PES University

June - Nov 2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

"Image Captioning web app using Flask and Machine Learning"

is a bonafide work carried out by

RAJ.NAGRANI	**PES2201800177**
MANIKYA.PRAKASH.SARASHETTY	**PES2201800301**
NANDINI.R.SONTH	**PES2201800401**
RAMYASHREE.J.R	**PES2201800728**

In partial fulfilment for the completion of seventh semester Capstone Project Phase - 2 (UE18CS390B) in the Program of Study -Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period June 2021 – Nov 2021. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 7th semester academic requirements in respect of project work.

Signature Prof.Ruhi.Dubey Assistant Profeesor	Signature Dr. Sandesh B J Chairperson	Signature Dr. B K Keshavan Dean of Faculty

External Viva

Name of the Examiners	**Signature with Date**
1. _____	_____
2. _____	_____

DECLARATION

We hereby declare that the Capstone Project Phase - 2 entitled **"Image captioning web app using Flask and Machine Learning"** has been carried out by us under the guidance of Prof. Ruhi Dubey, Assistant Professor and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester June – Nov 2021. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

PES2201800177	**RAJ.NAGRANI**
PES2201800301	**MANIKYA.PRAKASH.SARASHETTY**
PES2201800401	**NANDINI.R.SONTH**
PES2201800728	**RAMYASHREE.J.R**

ACKNOWLEDGEMENT

I would like to express my gratitude to Prof. Ruhi.Dubey, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE18CS390B - Capstone Project Phase – 2.

I am grateful to the Capstone Project Coordinator, Dr.Sarasvathi V, Associate Professor, for organizing, managing, and helping with the entire process.

I take this opportunity to thank Dr. Sandesh B J, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department. I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice- Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

Our main goal is to use machine learning techniques to create and construct an image captioning system. Creating descriptions for photos has always been an interesting topic of computer vision. For the image captioning in this project, we will use Neural Networks. The encoder is a Convolution Neural Network (ResNet), which accesses picture features, and the decoder is a Recurrent Neural Network (Long Short Term Memory), which generates captions for the images using the image features and vocabulary that has been constructed. Image captioning is extremely useful in a variety of applications, including analysing massive volumes of unlabeled photos and detecting hidden patterns for Machine Learning applications such as directing self-driving cars and developing software to assist blind people.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Image captioning used to be a very difficult operation, and the captions that were created for the input image were often irrelevant. Many tasks that were tough and difficult to perform became easier with the evolution of Machine Learning,Deep Learning and Neural Networks, as well as text processing techniques like Natural Language Processing. These are extremely useful in picture identification, image categorization, image captioning, and a variety of other AI applications. Image captioning is the process of creating descriptions for what is happening in an image. Basically ,this model takes image as input and gives caption for it. With the advancement of the technology the efficiency of image caption generation is also increasing. This Image Captioning is extremely beneficial for a variety of applications, including the increasingly popular self-driving cars. Many Machine Learning tasks for Recommendation Systems can benefit from image captioning. Many models for picture captioning have been proposed, including object identification models, visual attention-based image captioning, and Image Captioning with Deep Learning. There are a variety of deep learning models available, including the Inception model, the VGG model, the ResNet-LSTM model, and the conventional CNN-RNN model. We will describe the model we used for captioning the photos in this paper. Specifically, the ResNet-LSTM model. The amount of memory available on the GPUs used to train the network, as well as the length of training time permitted, dictate the neural network's constraints. With only 6000 photos used to train the model, our network takes about one day on Google Collab GPUs.Using a huge dataset like MS-coco, we can improve the outcomes, according to our findings. Our Flask web program accepts an image as an input from a web page running on the localhost development server, processes it, and outputs an image with a caption and audio. The goal of producing audio is to convert this app into a flutter android app that blind people may use to help with future project upgrades.

CHAPTER 2

PROBLEM STATEMENT

Image caption is a key aspect of scene understanding, which combines computer vision and natural language processing knowledge to automatically generate natural language descriptions based on the content observed in an image. When compared to human-generated captions, the developed model was able to generate captions that were somewhat equivalent. Using the image retrieved, we attempted to create a user-friendly web application capable of generating captions and, with the option of a narration.

CHAPTER 3
LITERATURE REVIEW

1] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, Yoshua Bengio; Proceedings of the 32nd International Conference on Machine Learning, PMLR 37:2048-2057, 2015._The paper's main goal in generating captions was the use of techniques of neural networks and visual attention. Only the most important features were supplied to RNN. The advantage of this method was beforehand, all of the noise and clutter in the real-world image was ignored (before feeding into the RNN). Limitations of the this paper include even terms like 'a,the,of' were given attention while attending to the image, despite the fact that they contain no visual signal in the image.

2] Farhadi A. et al. (2010) Every Picture Tells a Story: Generating Sentences from Images. In: Daniilidis K., Maragos P.,Paragios N. (eds) Computer Vision – ECCV 2010. ECCV 2010. Lecture Notes in Computer Science, vol 6314. Springer,Berlin, Heidelberg. This was one of the first papers to be out in this Image Captioining field.In this to define an image 3 spaces were considered namely-Image Space,Sentence space,Meaning space. Image space and Sentence space was mapped to a meaning space in triplet form as <image,action,object>,which best described an image. If an image and sentence have high level of similarity then they will be mapped to a triplet. Laid the foundation for Image captioning tasks. A middle meaning space was missing[it was required in case of a less score between an image and sentence.The results from this method had low accuracy.

3] O. Vinyals, A. Toshev, S. Bengio and D. Erhan, "Show and Tell: Lessons Learned MSCOCO Image Captioning Challenge," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 652-663, 1 April 2017.CNN and RNN for image captioning tasks. Convolutional neural networks were used to extract features from the images. So, CNN acts as an encoder, first for classification of tasks and the last layer's output is provided as the input to RNN.RNN acts as a decoder that generates sentences. LSTM networks (Long Short Term Memory) was the type of RNN used. Detectors were trained to extract various features out of the images and a translation model was trained on a set of captions to generate the appropriate descriptions for the image at hand.With the advent of CNN and RNN a high performance was achieved in this field and found applications in various fields of study.This method fed all the feature vector into the RNN(which included even the unnecessary features that were of no importance).

4] R Ani, Effy Maria, J Jameema Joyce, V Sakkaravarthy, MA Raja 2017 International Conference on Innovations in Green Energy and Healthcare Technologies (IGEHT), 1- 6, 2017.In this paper the strategy they used can assist the impaired people with perusing any printed text in vocal structure. A specs inbuilt camera is utilized to catch the text picture from the printed text and the caught picture is investigated utilizing Tesseract - Optical Character acknowledgment (OCR). The identified text is then changed over into discourse utilizing a reduced open source programming discourse synthesizer, eSpeak. Raspberry pi is the fundamental objective for execution in this system.It was one of the first tech to help visually impaired people on a large scale.This method fails even in the slightest of noise and can only work for printed text images.

5] Doshi, Text Reader for Visually Impaired Using Google Cloud Vision API, international journal of innovative research in technology (IJIRT). Vol. 4, 5/18. The Objective of paper is that the system extracts the text from images using google cloud vision API, which is capable of recognizing text in multiple conditions, the response obtained is in the form of the JSON structure. The text is obtained and then converted into speech using the TTS engine. Then the text is stored as an mp3 file and then played using an mp3 player finally the output text is converted into audio output in the form of synthetic speech.This method works even in the presence of blur, low resolution ,low contrast. This method works where OCR method fails. According to one survey done on Google cloud Vision API, if we add other noises(leaving blur, low contrast and low resolution),the results of Caption can vary differently.

6] In this paper, they mainly described the three image captioning methods which are CNN-RNN based, CNNCNN based, and Reinforcement-based models. They had given the representative works, evaluation metrics with benefits and challenges of the three systems.

7] In this paper, Aker and Gaizauskas represented an approach to automatic captioning of geo-tagged images by summarizing multiple web documents that contain image locations. They had a dependency pattern model which is provided by various scenarios, they used higher ROUGE scores than n-gram models which leads to dependency relational patterns.

8]In this overview, Horang Wang and the team has compiled all the aspects of the image caption generation task and had briefly discussed the model framework proposed in recent years to solve the tasks which mainly focuses on the algorithms of the various mechanisms and they had provided the method to use attention mechanism for the system. They had summarized the large datasets and the evaluation criteria which are commonly used in images caption generation methods.

9] Yang et al. proposed a system that does the automatic generation of a natural image, which helps to understand the images. They proposed a multimodal neural network model that consists of detecting the objects and does the module localization which is similar to the humans, and gets the image description's accordingly.

10] In this paper, they had developed a model using the concepts of a Long Short- Term Memory and Convolutional Neural Network model built an image captioning generator using CNN with LSTM. The CNN extracts the image vectors and LSTM extracts the word vectors

CHAPTER 4

PROJECT REQUIREMENTS SPECIFICATION

4.1. Product Perspective

The goal of automatic picture captioning is to automatically construct properly formed English phrases to explain the content of an image, which has a wide range of applications in areas such as virtual assistants, image indexing, editing programme recommendation, and impaired assistance . Although it is simple for a human to describe an image, it is quite complex for a machine to do so . Captioning an image requires not just detecting the items in the image but also capturing how these things are related to one another, their properties, and the activities they are engaging in. Furthermore, semantic knowledge should be communicated in natural language, which necessitates the development of a language model based on visual perception.

4.1.1 Product Features

This product will be performing the following actions:

1] **Object detection** – Users can select a picture from their system and upload it to the web app and the system will describe the picture in one sentence with identifying large objects. This will enable the users to get insights from their picture.

2] **Person detection** – Users can upload a picture of any human being and the system will be able to describe the human being with the features be like gender, age etc.,

3] **Audio output** – The caption can be heard by the audio.

4.2. User Classes and Characteristics

USERS - These users will not have access to alter anything in the web app. They can just comfortably use the features of the system with the valid internet connection.

ADMINS - These users have the access to change anything in the app environment and they can solve errors in the environment of the app. These people take care of maintainability of the app and will not check out the content of the non-admins.

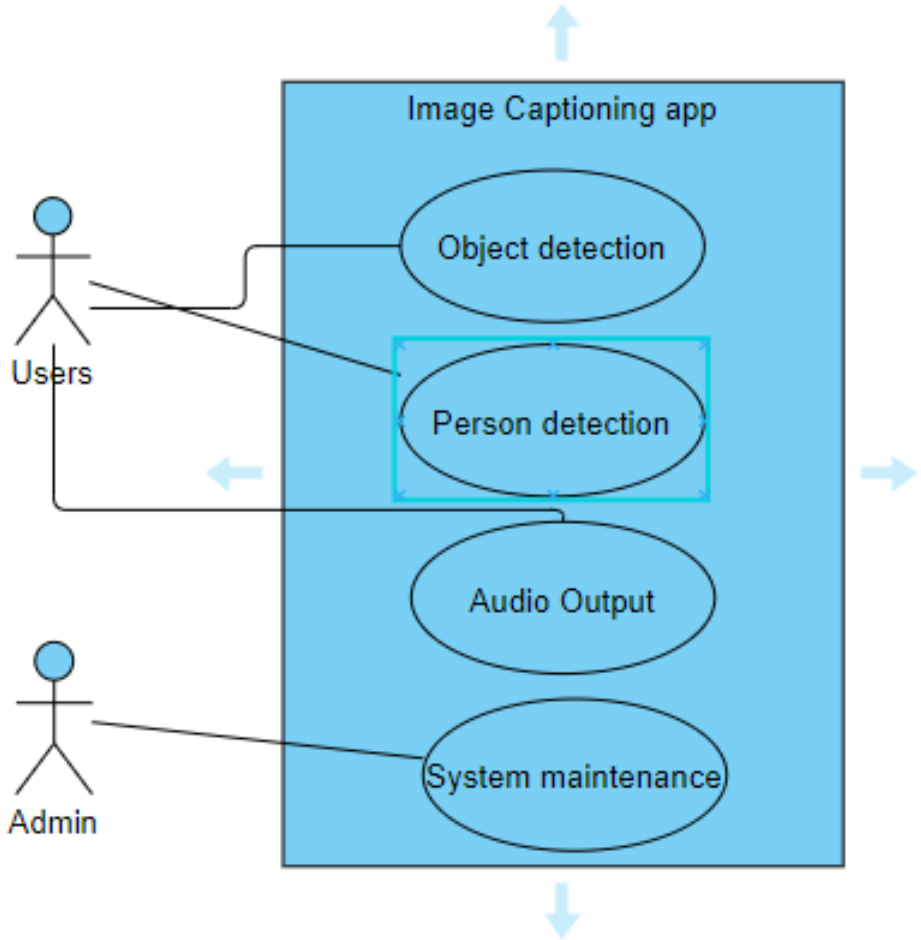


Fig.1.Usecase Diagram

4.3. Operating Environment

Flask is a Python-based web application framework. It was created by Armin Ronacher, who is the founder of Pocco, an international group of Python enthusiasts. The Werkzeug WSGI toolkit and the Jinja2 template engine are the foundations of Flask. Both are Pocco initiatives. Flask is a compact and lightweight Python web framework that provides essential tools and capabilities for building online applications in Python. Because you can build a web application quickly using only a single Python file, it gives developers more flexibility and is a more accessible framework for new developers. FLask is a prototype that is used to construct instances of web apps, or web applications to put it another way. So, once we've imported Flask, we'll need to make a Flask class instance for our web app.

4.4. General Constraints

- **Time Frame** : In next 3 months project documentation for the design must be done. And the whole project must be completed by Dec 2021.
- **Data Migration** : Although the application is going to be prepared over from the scratch, it is also going to include some existing applications. These parts are needed be implemented and work correctly.
- **Resources** : As a beginning we are going to implement the application in our own device and check the system to see if they are working as intended. As platform we are going to use Flask and simple HTML.
- **Peer/User Review** : It is important that the project is examined by a possible user to see the design mistakes and possible problems better and make a shortage.

Assumptions and Dependencies

- Users must have valid internet connection.
- This system is operated in all higher versions of browser which support Flask and Jinja.

4.5. Risks

- The concern is detecting objects correctly and conveying that information to users through voice-over system. This needs to work accurately also so that users do not get the incorrect information about the picture.
- The audio may also be wrong sometimes.

4.6. Functional Requirements

Any system must have important features enabled in it to perform smooth functioning of the project. Some of which include the following:

4.6.1 Object detection

Description and Priority

It is a high priority task and in which the user gets a voice for the uploaded picture in web app.

Stimulus/Response Sequences

This neural system for image captioning uses an image as input, and the output is a voice command describing the visual content of a picture by identifying the objects.

Functional Requirements

This project was built using a convolutional neural network (CNN) to extract the visual features, and uses a recurrent neural network (RNN) to translate this data into text. Both CNN and RNN parts can be further trained using the Tensor Flow library. The text can be converted to speech using available APIs

4.6.2 Person Recognition

Description and Priority.

It is a high priority task and in which the user gets voice commands for the uploaded picture in system with human beings as objects.

Stimulus/Response Sequences

This system uses an image with text as input and starts giving voice response line by line going through the text in the uploaded picture.

Functional requirements

The encoder is a Convolution Neural Network (ResNet), which accesses picture features, and the decoder is a Recurrent Neural Network (Long Short Term Memory), which generates captions for the images using the image features and vocabulary that has been constructed.

4.6.3 Audio Output

Description and Priority

It is a high priority task and in which the user gets the audio of the generated caption.

Stimulus/Response Sequences

This system uses data input and provide the audio of the caption.

Functional Requirements

Here we use the Javascript to generate the audio in the frontend Html page using Speech synthesis function. The Speechsynthesis interface of the web speech API represents a speech request. It contains the content the speech service should read and information about how to read it (e.g. language, pitch and volume.)

4.7 External Interface Requirements

4.7.1 User Interfaces

The following section gives the overview of the application's user interface. A user interface is where the user interacts with the application. The overall flow of the application and the modules will be explained in this section.

Home Screen

The home screen is the first screen that the user visits upon opening of the application. In this screen, the application provides an option to select the image from the local system and button to submit.

Navigation to Output Page

After the submit button the web page is navigated to predict.html where it will display image, caption and narrate the caption.

4.7.2 Hardware Requirements

- Microsoft Windows 7/8/10 (32 or 64 bit)
- 2 GB RAM minimum, 8 GB recommended
- 2 GB of available disk space minimum, and minimum 4 GB recommended
- 1280 x 800 minimum screen resolution

4.7.3 Software Requirements

- Python
- Flask
- Visual studio IDE
- Jinja extensions

4.7.4Communication Interfaces

- HTTP/HTTPS - used to perform basic communications for data to be transferred.
- A speech synthesiser is a digital device that accepts input, processes data, and produces audible language, in addition to providing standard organisation availability. Any text, predetermined input, or controlled nonverbal bodily movement can be translated into audible speech.

4.8 Non-Functional Requirements

4.8.1 Performance Requirements

- Reliability - This web application is reliable to perform the required functions within a time span designed as it considers static objects .
- Robustness - User will get response from the website ,even when software doesn't recognize the inputs given by the user.
- Availability - Since we need internet access for this application to work with, any network issues/failures may lead to improper functionality of the web app.

4.8.2 Safety Requirements

- This app cannot guarantee that the predicted output will be completely accurate.
- Users should give access to only those features they intend to use before starting to use the web app.

4.8.3 Security Requirements

- The web app Security ensures that the content of the clicked pictures will be safe and it will not be viewed by others.
- The admins will be taking care of any technical errors in the app.

4.9 Other Requirements

- **Maintainability**: If there's a network issue while in the process of processing an image in the app, the user can close and restart the server of web application to restore the same state as before.

CHAPTER 5

SYSTEM DESIGN

5.1Design Considerations

5.1.1 Design Goals

Looking at the other researches that is done in the similar filed our web app can detect objects in the images and it does not except any other form of input like videos

,therefore increasing the efficiency and providing considerably accurate results to the users. Our web app provides easy to use interface, as the user just needs to upload the image. All the analysis will be done in the backend quickly as possible and the user can benefit from the available results.

Our app service is available 24/7 or whenever required by the user. Our application analysis speed is quite fast provided with good internet connection.

5.2 Architecture Choices

We chose Flask over other open source backend technologies such as Java, Ruby, Perl, and others. Flask comes with several basic features and lets developers to add as many libraries or plugins as they like to an extension. Flask should be your first pick if you have a simple, novel use case that needs to be added to an existing application. We considered using Flask for our web application. Earlier models merged nouns, adjectives, and other words to build a sentence by using information about visual positions, predicting motion in the image, and searching for nouns, adjectives, and other words. Traditional models all have the same flaw: they don't make feature observations on the image or perform actions on the image. In the discipline of image representation, the display of neuronal organisations has made significant progress.

Advantages

- The advantages of using Resnet is doesn't raise the training error percentage and networks with a large number of layers (even thousands) can be easily trained.
- Using identity mapping, ResNets can aid with the vanishing gradient problem.
- There are numerous advantages to using a flask, include:
- Compatibility with the most recent technology.
- Experimentation with technology.
- For simple circumstances, it's easier to utilise.
- The codebase is relatively modest.
- For simple apps, there is a lot of scalability.
- It's simple to create a quick prototype.
- Routing URLs is simple.
- Applications are simple to create and manage.

Disadvantages:

- It only works when we have good network/internet connection.
- One concern is to get the most accurate results.
- No login or authentication in Flask and not suitable for big applications.

5.3 Constraints, Assumptions and Dependencies

- Performance related issues: Due to network dependency of our app, if there are network issues the system cannot deliver voice in real time.
- All the backend analysis is done by implemented model using Resnet. We will develop our web app using flask, which would be available all the time for use and hence highly reliable and this is the development server which can run in local machine. Further enhancements can be deployment to cloud servers and developing android applications.

Hardware or software environment:

Hardware environment-

- Microsoft Windows 7/8/10 (32 or 64 bit)
- 2 GB RAM minimum, 8 GB recommended
- 2 GB of available disk space minimum, 4 GB recommended
- 1280 x 800 minimum screen resolution

Software environment-

- Python
- Flask
- Visual studio IDE
- Jinja extensions

- Deployment in target environment: Network issues could be a problem.
- **Maintainability**: If there's a network issue while in the process of processing an image in the web app, the user can close and restart the server to upload the picture.

5.4 High Level System Design

The logical user groups are the people who want to gain insights or summary from the photo.The data components are dataset used for training and deployment. The application components are the web application to show the integrated project and Flask as the backend. The interfacing system will be the web application built using HTML,CSS and Javascript. Dependencies and modules required by Flask integrated in the requirements.txt file. Relevant design procedures and patterns will be used to make the web application integrating all the required interfaces for the proper functioning of the product.

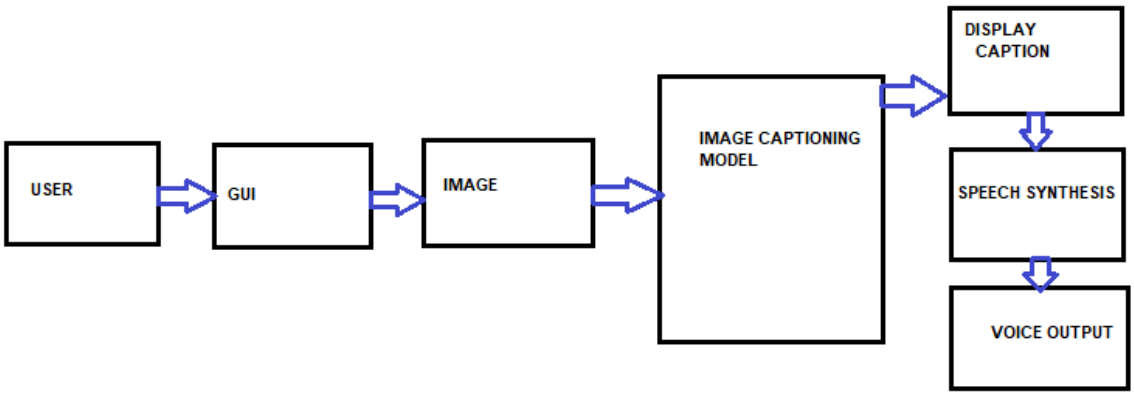


Fig.2.Control Flow

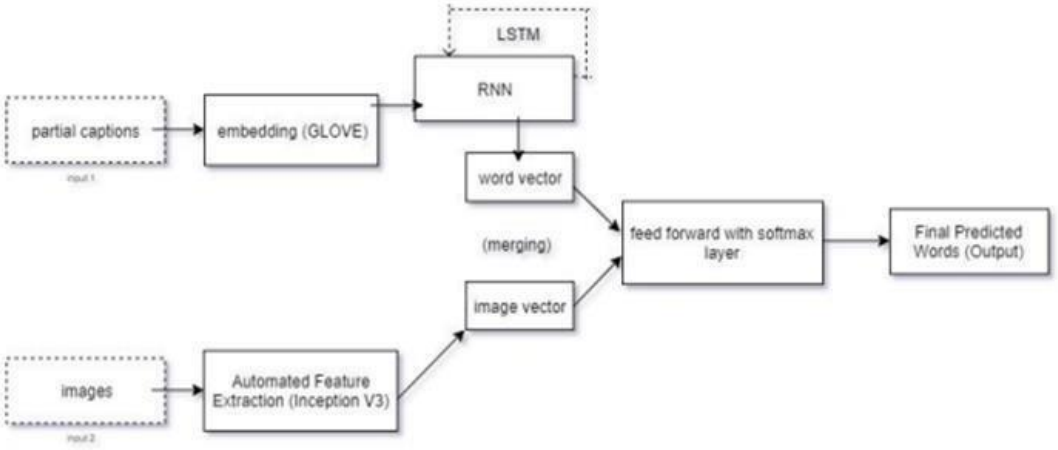


Fig.3.System Architecture(High level design)

The high-level design has these main components:

- **Start the Flask server in the local Machine**

  The user will download all the python and html files and start the Flask server in the VS code in the local machine.

- **Image Upload**

  After the server starts user will have the option to upload image from the system which is then given as input to the model to obtain results(speech).

- **Model Training and Results Obtained**

  The model will then use the training dataset to train itself using CNN and RNN neural networks. The results shown will be text i.e captions suitable to the image provided which is again converted to speech as the final output. There will be modules for cleaning, pre-processing, model building, model training, result generation, result display.

5.5 Design Description

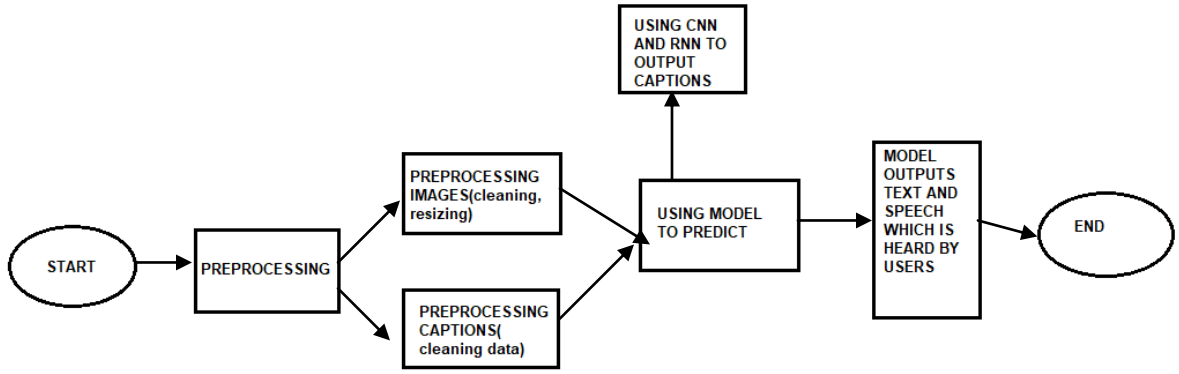


Fig.4.Workflow Diagram

5.5.1 Master Class Diagram

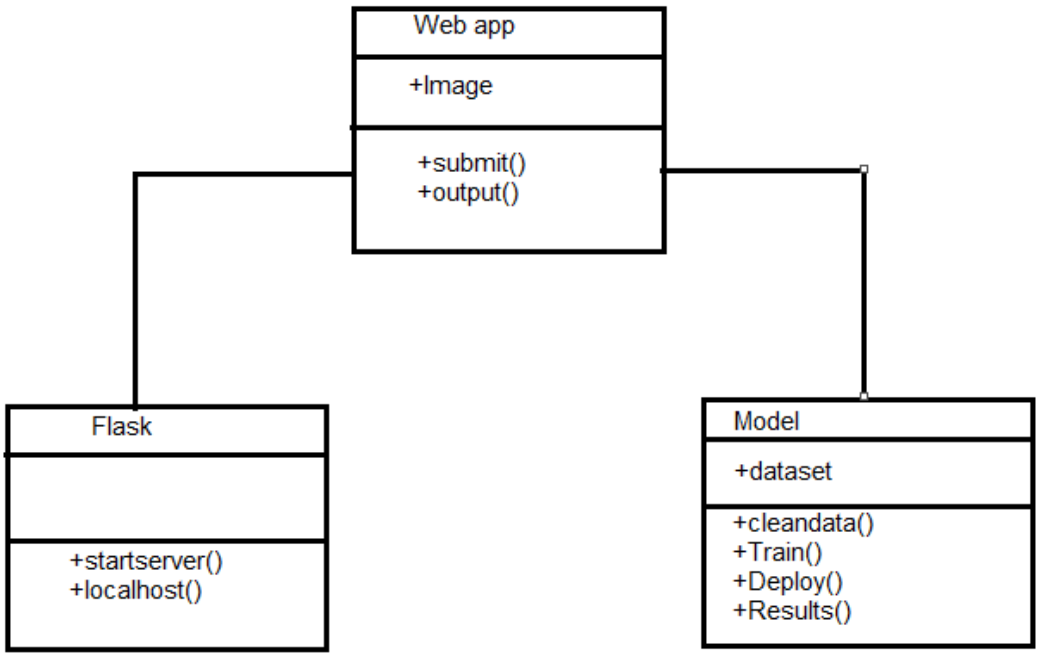


Fig.5.Master class Diagram

5.5.2 Activity Diagram

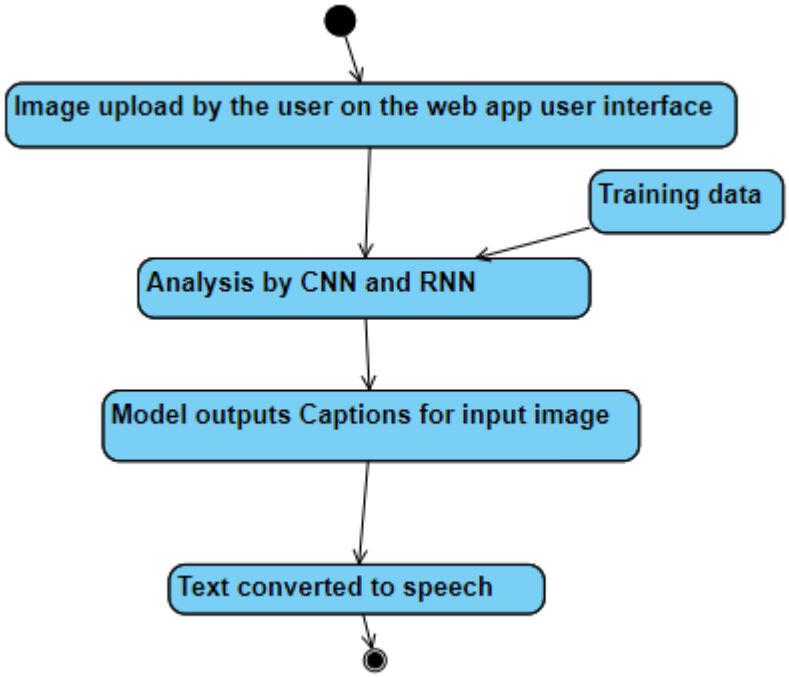


Fig.6.Activity Diagram

5.5.3 ER Diagram

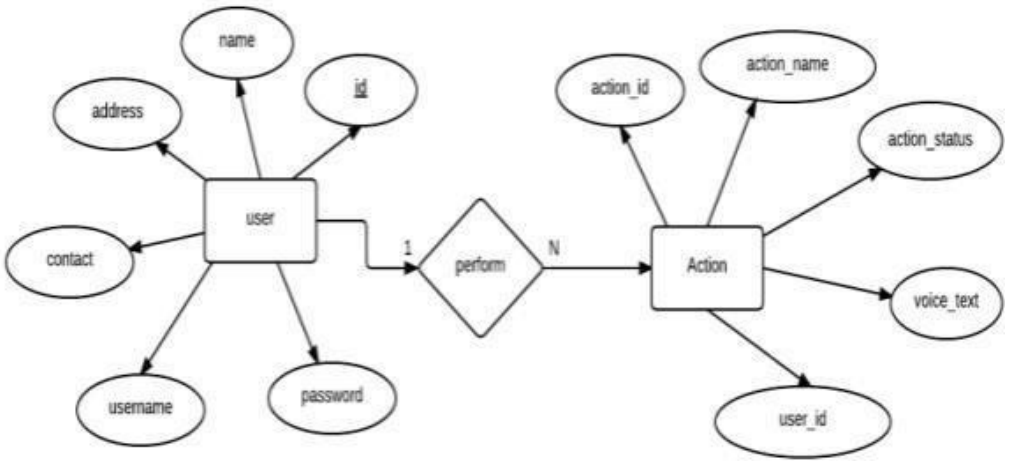


Fig.7. ER Diagram

#	Entity	Name	Definition	Type
		ENTITIES		
3.	Result	-	Output Display	Char
		DATA ELEMENTS		
1.	Image	-	Input by user	Image
2.	Training Data	-	Training data for analysis	ModelData
3.	Speech	-	Output Format	Audio

Table.1.Entities and elements

5.5.4 Class Diagram

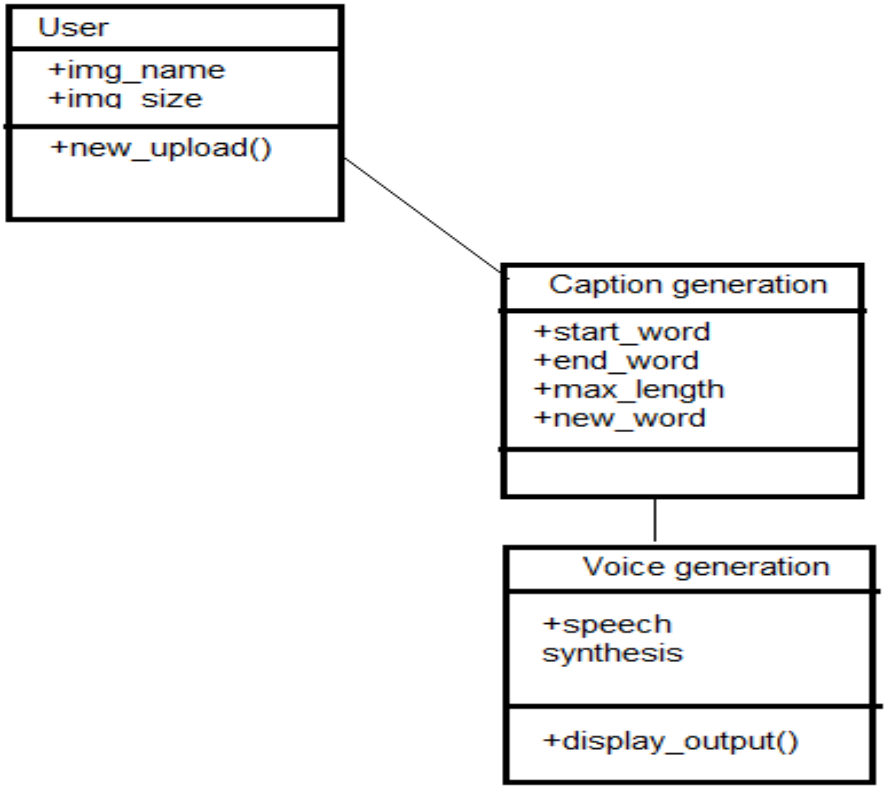


Fig.8. Class Diagram

5.5.5 Sequence diagram

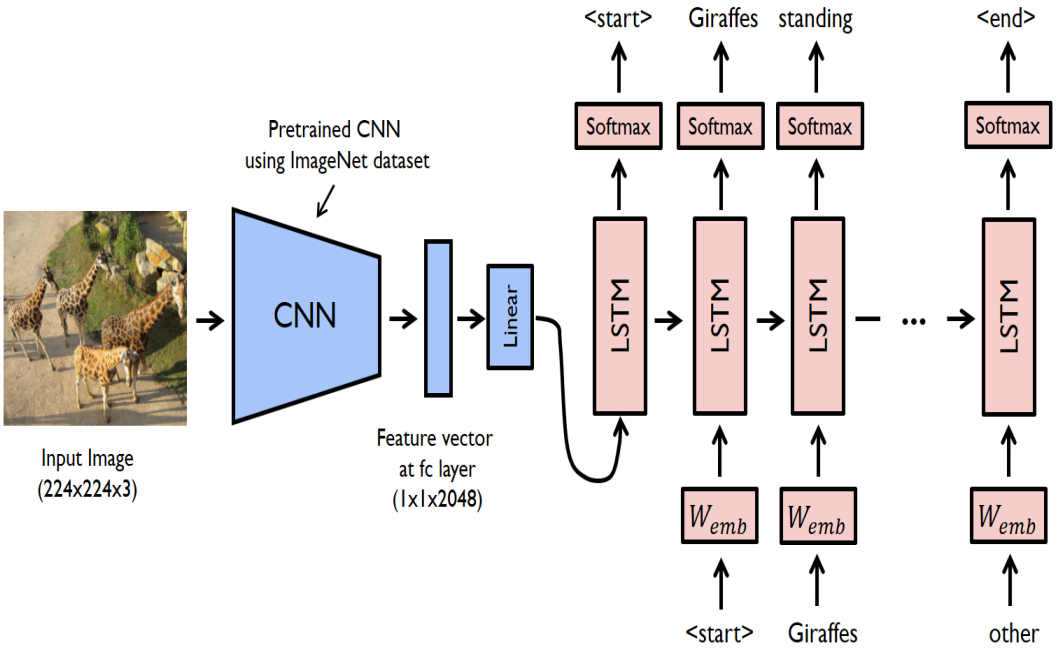


Fig.9.Sequence Diagram

5.5.6 Merging Architecture

In this Image is not introduced into RNN ,both the inputs are encoded separately.

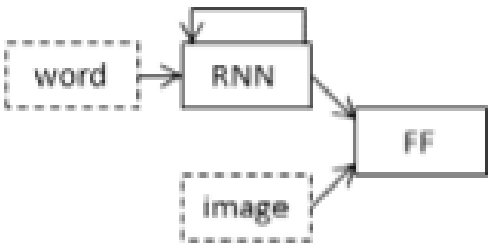


Fig.10.Merging Architecture

5.6 External Interface

5.6.1 User Interface

Web app will be having an option to upload.

User selects the image from the system and upload the picture and submit.

Page will be navigated to predict page displaying caption and voice.

5.6.2 Hardware Interface

Laptop or pc to run the flask program.

5.6.3 Software Interface

The system must be connected to the internet.

Web application(any browser).

5.6.4 Communication Interface

HTTP for get requests.

CHAPTER 6

PROPOSED METHODOLOGY

There is a vanishing gradient problem when utilising the typical CNN-RNN model, which makes it difficult for the Recurrent Neural Network to learn and be efficiently trained. So, in order to alleviate the gradient descent problem, we propose this model in this study in order to improve the efficiency of producing captions for images and to reduce the number of captions generated.

Caption accuracy should be improved. In this work, we will explain how to use the Resnet-LSTM model for image captioning. Encoding is done with Resnet Architecture, and decoding is done with LSTMs. We will now train the model with these two parameters as input once the image is given to Resnet (Residual Neural Network), which extracts the image features with the help of vocabulary that is constructed using training captions data. Following the training,we will test the model.

6.1 COLLECTION OF DATA SET

Many data sets, such as ImageNet, COCO, FLICKR 8K, and FLICK 30K, can be utilised to train a deep learning model for generating captions for images.

The FLICKR 8K data set is used to train the model in this paper. For training the Image Caption Generating Deep Learning Model, the FLICKR 8K data set works well. The FLICKR 8K data set contains 8000 photos, with 6000 images suitable for deep learning model training, 1000 images for development, and 1000 images for testing model. The Flickr Text data collection contains five descriptions for each image, each of which describes the actions depicted in the image.

6.2 PREPROCESSING OF IMAGES

We must preprocess the photos after loading the data sets in order to use them as input to the ResNet. We must resize each image to the same size, 224X224X3, because we cannot feed different sized photos through the Convolution layer like ResNet. We're also converting the photos to RGB using the cv2 library's built-in capabilities.

6.3 BUILDING A VOCABULARY

We can't just feed the string captions to the neural network because it can't process strings, so we have to convert the string captions to numbers and develop a number vocabulary. This is referred to as caption encoding. To begin, we must build a new space in which all words in each caption are taken after preprocessing the captions provided in the training data set. Now we must assign numbers to the words in the order they appear in the dictionary. This area is now known as the vocabulary library. We will number each caption with the help of this vocab library by numbering their terms according to the vocab library. Each word in a caption is numbered by reference to its value in a previously created vocab library.

6.4 IMPLEMENTATION OF MODEL

After gathering the data and preparing the photos and descriptions, as well as developing vocabulary. We must now define the model for caption generation. ResNet-LSTM (Residual Neural Network-Long Short Term Memory) is the model we propose. Resnet is utilised as an encoder in this model, extracting image characteristics from photos and converting them to a single layered vector, which is then passed as input to LSTMs. Long Short Term Memory is employed as a decoder, with image attributes as input and a vocabulary dictionary for progressively generating each word of the caption.

6.4.1 RESNET

Using deep neural networks like RESNET (Residual Neural Network), which is a pretrained model for many image recognition and classification, became easier with the introduction of transfer learning (using knowledge gained in a training network on one type of problem and applying the knowledge to another problem of the same pattern). Because ResNet is a pretrained model on the ImageNet data set to identify the photos, we utilise it instead of the Deep Convolutional Neural Network. As a result, we may reduce the cost of computation and training time by employing the concept of transfer learning. If we had used a CNN that had not been pre-trained, the calculation cost would have gone up, and the model would have taken longer to learn. We can improve the model's accuracy by utilising a ResNet pretrained model.

Resnet50 is made up of 50 layers of deep convolutional neural networks. ResNet50 is the architecture of the Convolutional Neural Network that we use in our Deep Learning Model for Image Caption Generation. Because we don't need classification output in this research, the last layer of Restnet50 is omitted. Instead, we access the output of the o layer before the last one in order to get the image features as a single layered vector output. ResNet is recommended over typical deep convolutional neural networks because it comprises residual blocks with skip connections, which lessen the vanishing gradient problem in CNN. ResNet also reduces input feature loss when compared to CNN. When compared to traditional CNN, ResNet has greater performance and accuracy in image classification and extracting image features.,VGG.

Convolutional Layer, ReLU (Rectified Linear Unit) Layer, and Pooling Layer make up a standard CNN. The following is the output after processing the input through a typical CNN:

H(x)=f(wx+b) or H(x)=f(x), where H(x) is the output value, x is the input, w is the multiplied weights, b is the added bias, and f() is the activation function. In the case of classic CNN, we can see that input does not equal output. As a result, if we use this to extract picture features or classify photos, the results will be erroneous, and the accuracy will be low.

When it comes to the ResNet model, skip connections are at its heart. This skip connections are the shortcuts that the gradient takes to get to the output layer. The output is equivalent to the input when these skip connections are used. i.e., H(x)=x+f(x), where f(x)=0, as shown in the diagram. As a result, we can see that when the photos are fed into the ResNet model, the output is the same as the input, with no bias or weights added. As a result, when ResNet is used to extract picture features, there is little loss of data or image features. As a result, ResNet outperforms the classic CNN model in retrieving visual information.

Convolution, ReLU (Rectified Liner Unit), Batch Normalization, Pooling layer, and Flatten are some of the layers of the Residual Neural Network.

The following is a description of the various layers of a Residual Neural Network and how they work:

Layer of Convolution, After passing the image through the convolution layer, it is transformed to pixel values. On the image, image filters (feature maps) are applied, and convolution is conducted. The ReLU layer receives the output of this convolution layer. When the Rectified Liner Unit layer receives the convoluted picture matrix. It alters the pixel values by using the ReLU activation function.

The output from the ReLU layer is given to the Batch Normalization layer, which executes normalisation and standardisation operations on the network by adding extra layers to scale the input to a common size, resulting in a speedier and more stable network. This layer's output is then transferred to the pooling layer.

The pooling layer is commonly used to reduce the size of an image. The Max Pooling layer usually traverses a tiny matrix window over its input. It simply chooses the highest value in each submatrix. As a result, the size of the input matrix is reduced without much loss.

Layer should be flattened In this ResNet, the flatten layer's general function is to transform the picture featured matrix to a single layered vector. After applying the max pool function to the matrix, we wish to reduce the size of the matrix and transform it to a single layered feature vector with picture features. For this reason, flattening is performed immediately after max pooling in ResNet. After the matrix is transformed to a single layer, image features are extracted and given to the Long Short Term Memory Unit, which generates each term in the caption sequence using the vocabulary we've created.

As a result, we extract picture features from the ReNet model as a single layered vector, which is then sent to the LSTM Networks for caption generation.

6.4.2 LSTM

Long Short Term Memory Networks are used to construct captions utilising the output of ResNet (Image feature vector) and vocabulary built using training data set captions. When we give the first layer of the LSTM image feature vector and vocabulary as input, it constructs the first word of the caption using training knowledge. The image feature vector and previously generated words are used to generate the next words in a caption. Finally, the caption for the provided image is created by concatenating all of these terms.

Lengthy short term memory cells are the advanced RNNs which can remember info from long durations. Long Short Term Memory Networks can solve the vanishing gradient problem that Recurrent Neural Networks have. Due to the vanishing gradient problem, typical RNNs are unable to recall extended sequences of data. As a result, when it comes to caption production, RNNs are unable to recall crucial words that have been generated previously and are essential for the generation of future words. I am from France, for example, in the instance of anticipating the last word of this phrase. I am a native French speaker. It is critical to remember the first word France, which is impossible with standard RNNs, but not with Long Short Term Memory Networks. As a result, LSTMs are favoured over standard RNNs for caption production.

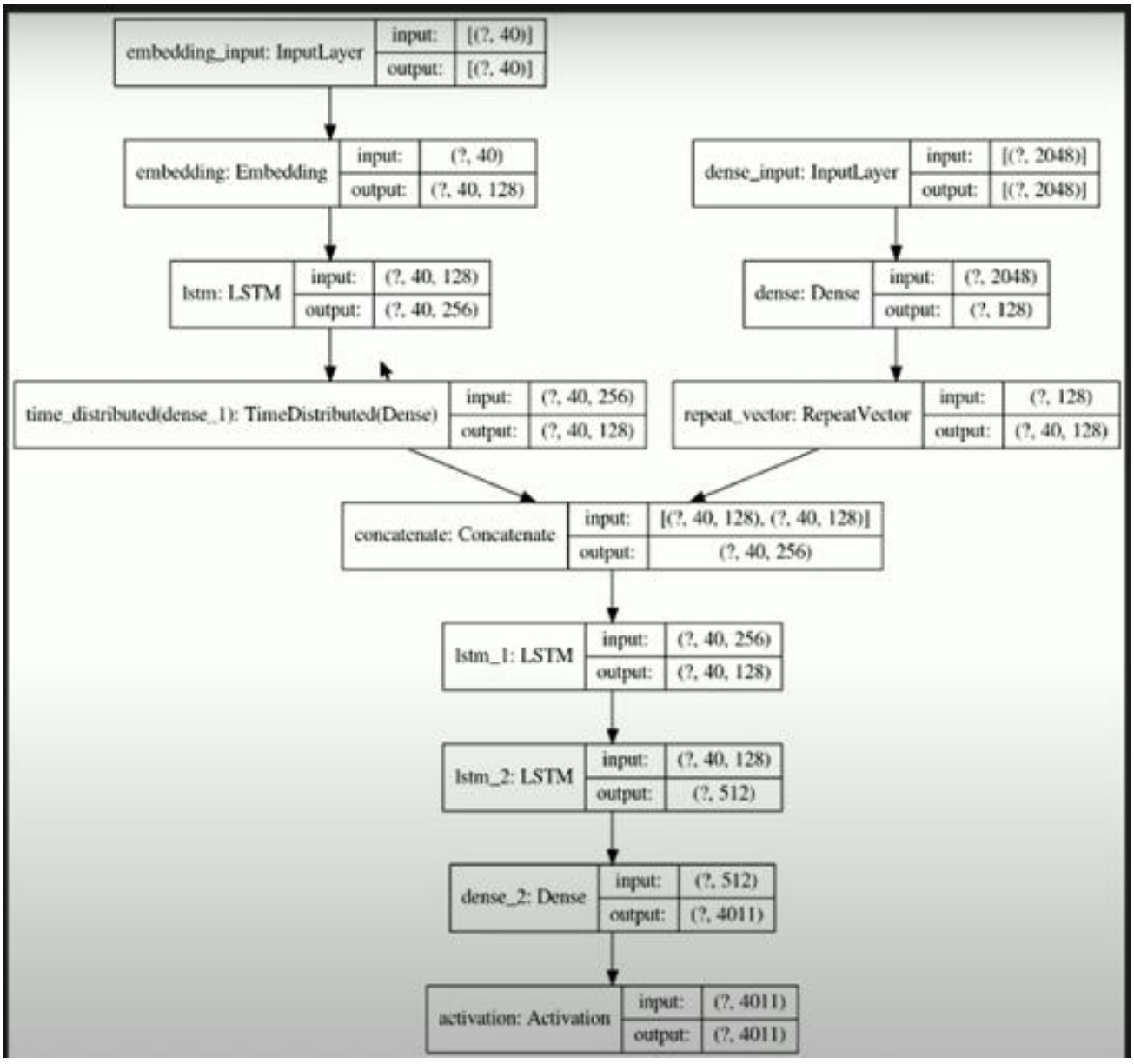


Fig.11 Architecture of the Model

CHAPTER 7

IMPLEMENTATION AND PSEUDOCODE

7.1 Image Preprocessing

In [ ]:
```
from keras.applications import ResNet50

incept_model = ResNet50(include_top=True)
```

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

In [ ]:
```
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)
```

7.2 Text preprocess

In [ ]:

```
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
```

7.3 Create Vocabulary

In [ ]:

```
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
```

7.4 Build generator function

In [ ]:

```
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
```

7.5 Model

In [ ]:

```
embedding_size = 128
max_len = MAX_LEN
vocab_size = len(new_dict)

image_model = Sequential()

image_model.add(Dense(embedding_size, input_shape=(2048,), activation='relu'))
image_model.add(RepeatVector(max_len))

image_model.summary()

language_model = Sequential()

language_model.add(Embedding(input_dim=vocab_size, output_dim=embedding_size, input_length=max_len))
language_model.add(LSTM(256, return_sequences=True))
language_model.add(TimeDistributed(Dense(embedding_size)))

language_model.summary()

conca = Concatenate()([image_model.output, language_model.output])
x = LSTM(128, return_sequences=True)(conca)
x = LSTM(512, return_sequences=False)(x)
x = Dense(vocab_size)(x)
out = Activation('softmax')(x)
model = Model(inputs=[image_model.input, language_model.input], outputs = out)

# model.load_weights("../input/model_weights.h5")
model.compile(loss='categorical_crossentropy', optimizer='RMSprop', metrics=['accuracy'])
model.summary()
```

7.6 Python and Html files

App.py

app.py

app.py > ...

```
70
71  app = Flask(__name__)
72
73  app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 1
74
75
76  @app.route('/')
77  def index():
78      return render_template('index.html')
79
80  @app.route('/after', methods=['GET', 'POST'])
81  def after():
82
83      global model, resnet, vocab, inv_vocab
84
85      img = request.files['file1']
86
87      img.save('static/file.jpg')
88
89      print("="*50)
90      print("IMAGE SAVED")
91
92
93
94      image = cv2.imread('static/file.jpg')
95      image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
96
97      image = cv2.resize(image, (224,224))
98
99      image = np.reshape(image, (1,224,224,3))
```

app.py

app.py > after

```
105
106     text_in = ['startofseq']
107     final = ''
108     print("="*50)
109     print("GETING Captions")
110
111     count = 0
112     while tqdm(count < 20):
113
114         count += 1
115
116         encoded = []
117         for i in text_in:
118             encoded.append(vocab[i])
119
120         padded = pad_sequences([encoded], maxlen=max_len, padding='post', truncating='post').reshape(1,max_len)
121
122         sampled_index = np.argmax(model.predict([incept, padded]))
123
124         sampled_word = inv_vocab[sampled_index]
125
126         if sampled_word != 'endofseq':
127             final = final + ' ' + sampled_word
128
129         text_in.append(sampled_word)
130
131     return render_template('predict.html', data=final)
132
133 if __name__ == "__main__":
134     app.run(debug=False,host='0.0.0.0')
```

Index.html

index.html

templates > index.html > html

```
<html>

<form action="{{url_for('after')}}" method='POST' enctype='multipart/form-data'>

        <h1>Image Captioning</h1>
        <br>
        <br>
        <input type="file" name='file1' accept="image/*">
        <br>
        <br>
        <br>
        <input type="submit">
    </form>
</html>

```

Predict.html

predict.html

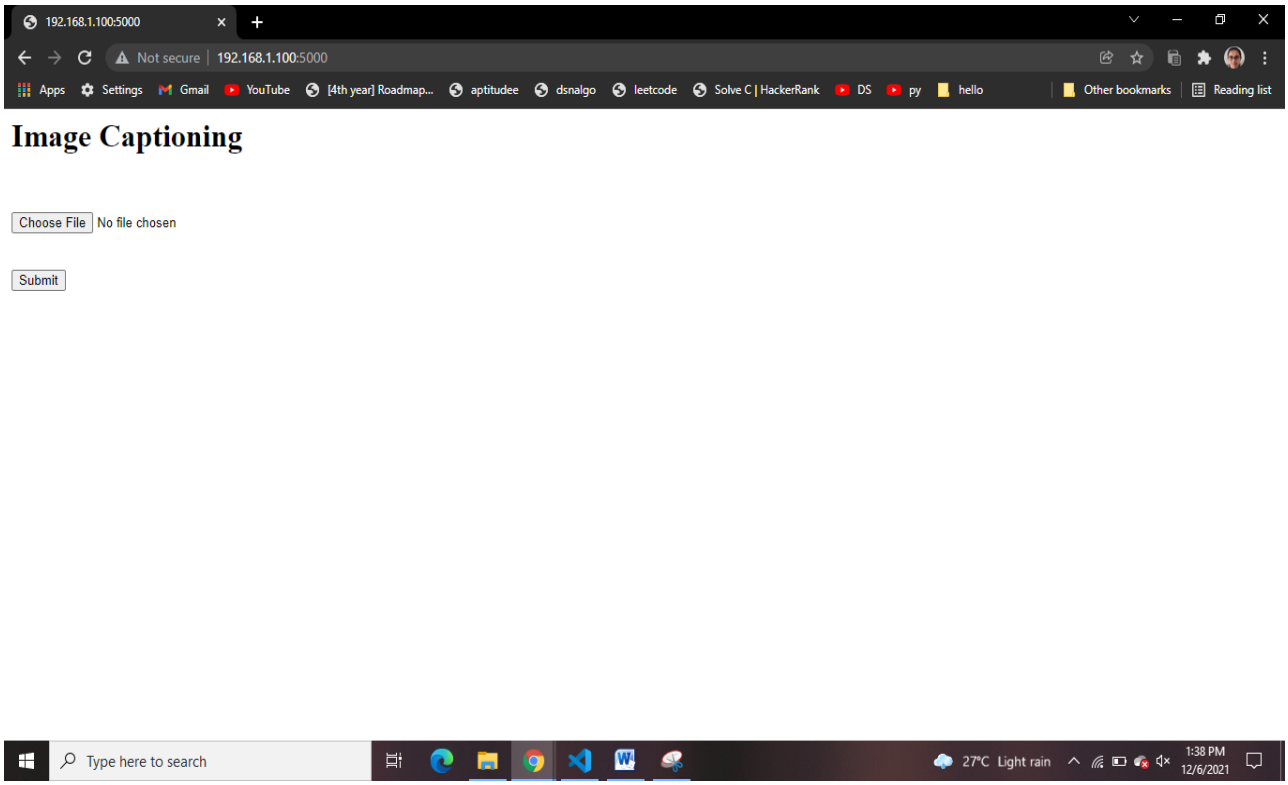
templates > predict.html > html

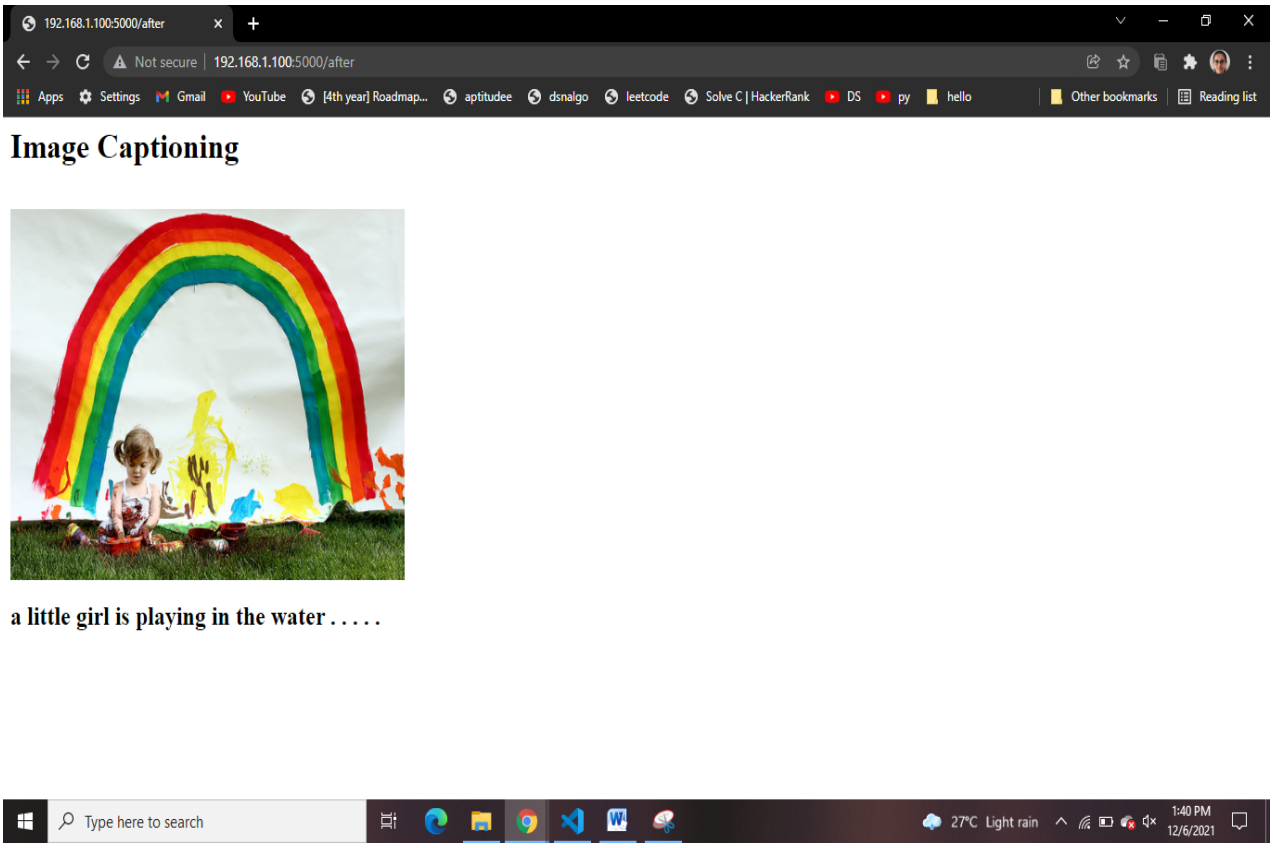
```
<html>
    <body>
        <h1>Image Captioning</h1>
        <br>
        <img src="{{url_for('static', filename='file.jpg')}}" alt="image" height=340px>


        <h2 class='text-white'>
            {{data}}
        </h2>

        <script type="text/javascript">
            var a = "{{data}}";
            var msg = new SpeechSynthesisUtterance();
            msg.text = a;
            window.speechSynthesis.speak(msg);
        </script>

    </body>


</html>
```

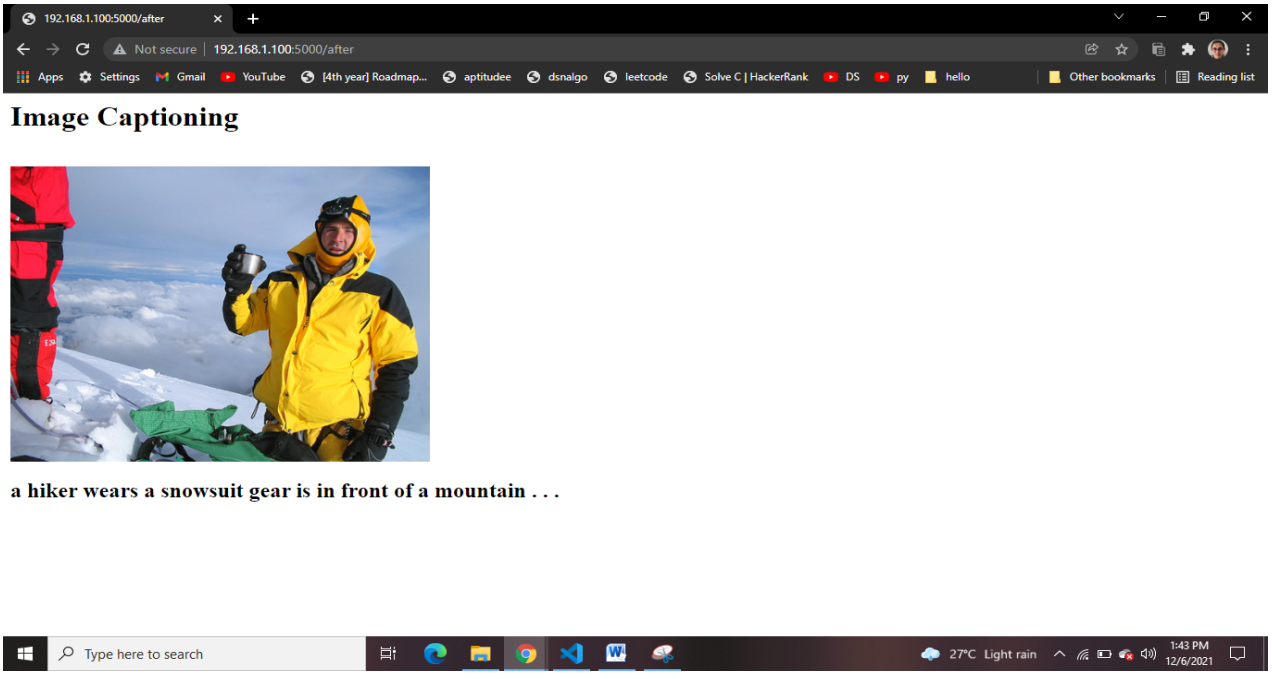
CHAPTER 8

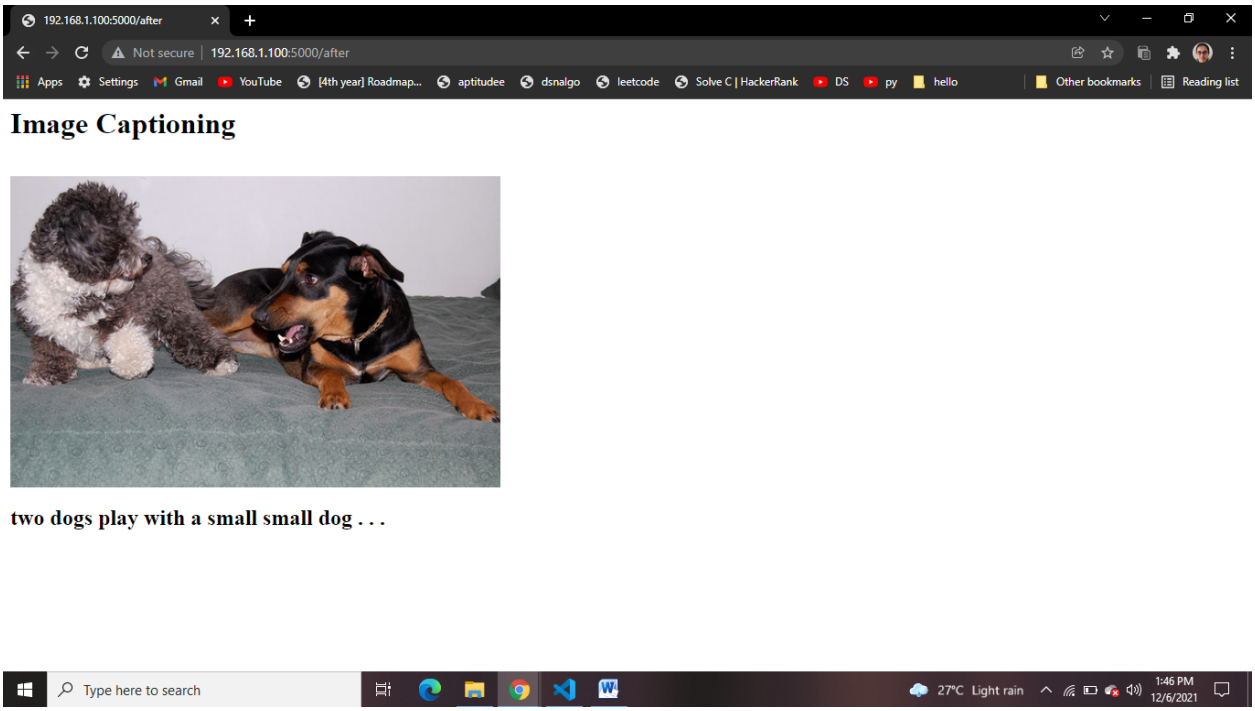
RESULTS AND DISCUSSION

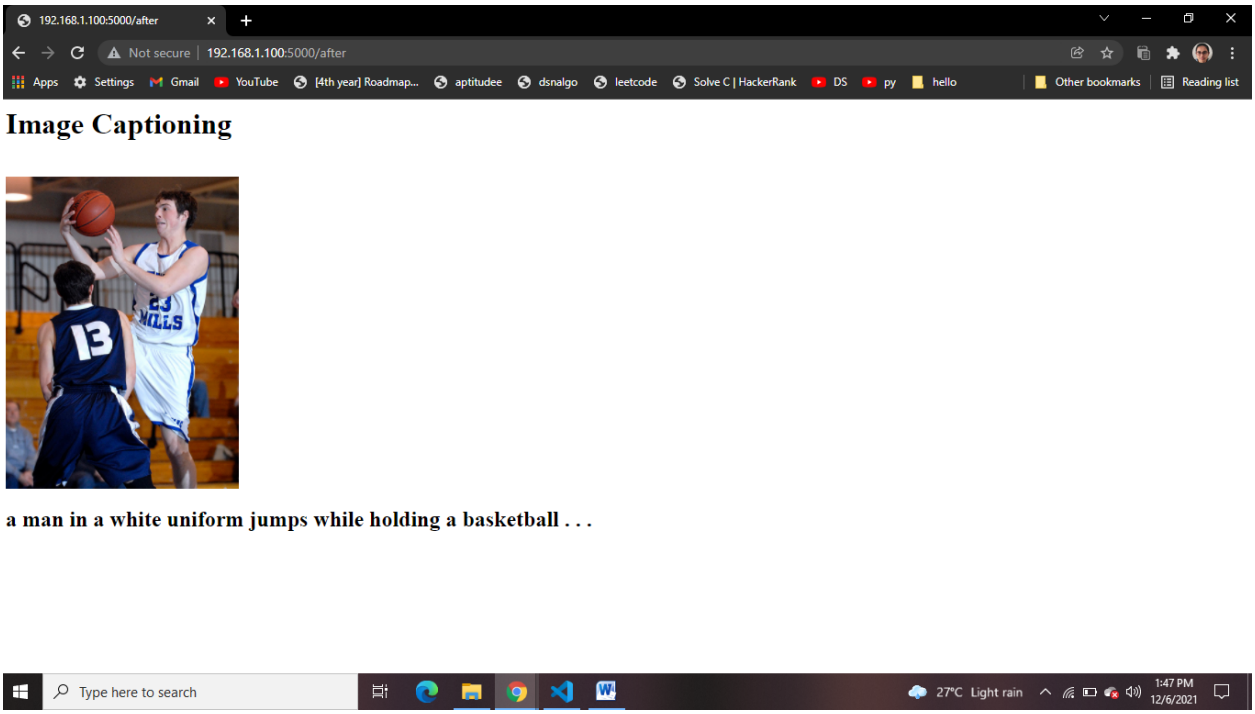
RESULTS

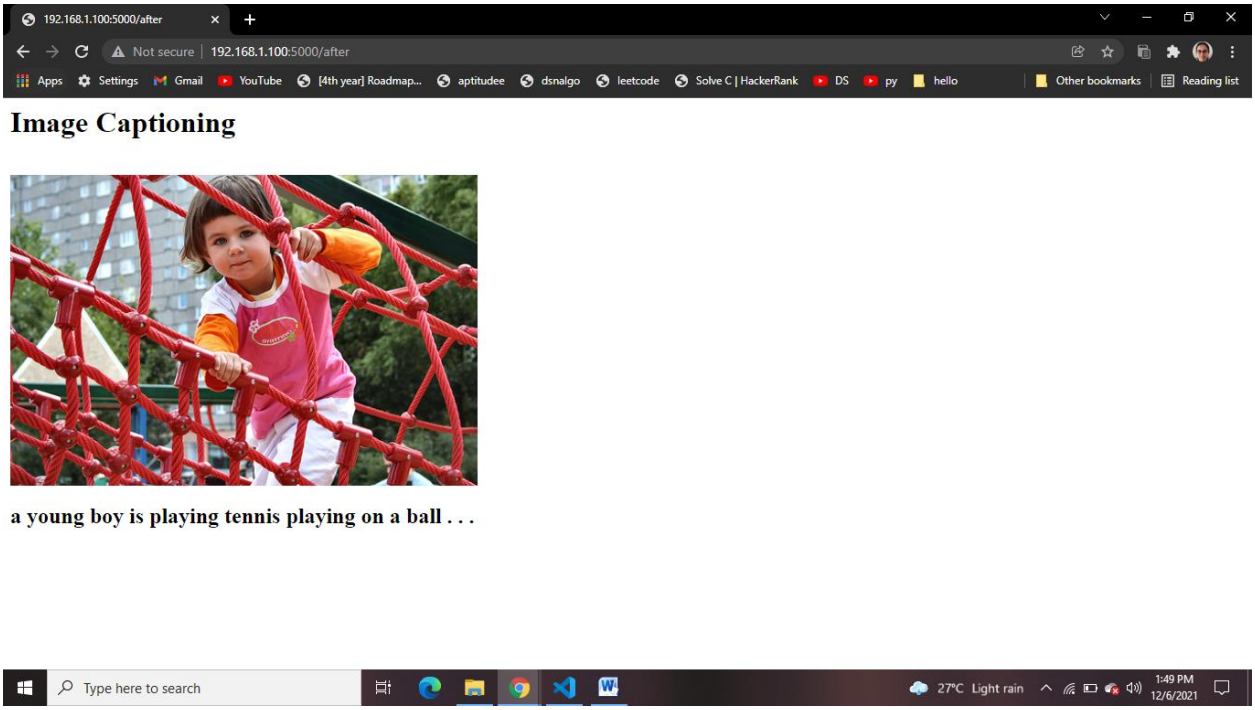


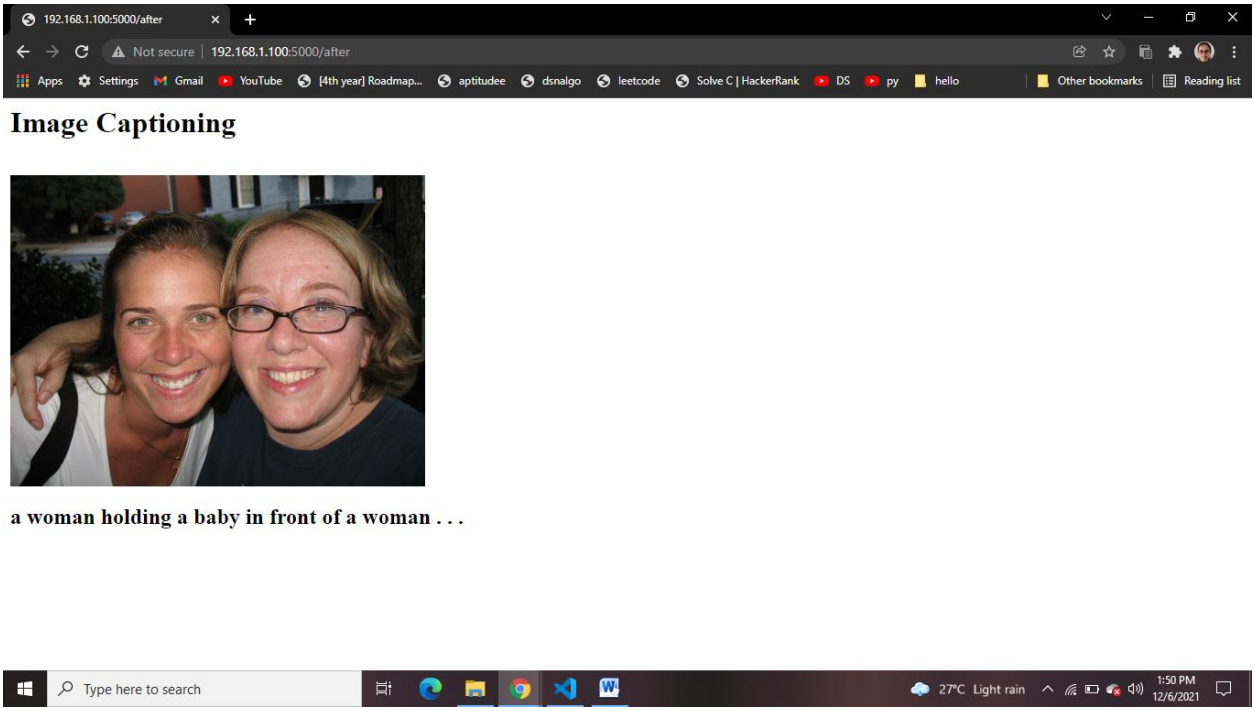

192.168.1.100:5000/after
Not secure
192.168.1.100:5000/after
Apps
Settings
Gmail
YouTube
[4th year] Roadmap...
aptitudee
dsnalgo
leetcode
Solve C | HackerRank
DS
py
hello
Other bookmarks
Reading list
Image Captioning
a little girl is playing in the water . . . . .
Type here to search
27°C Light rain
1:40 PM
12/6/2021

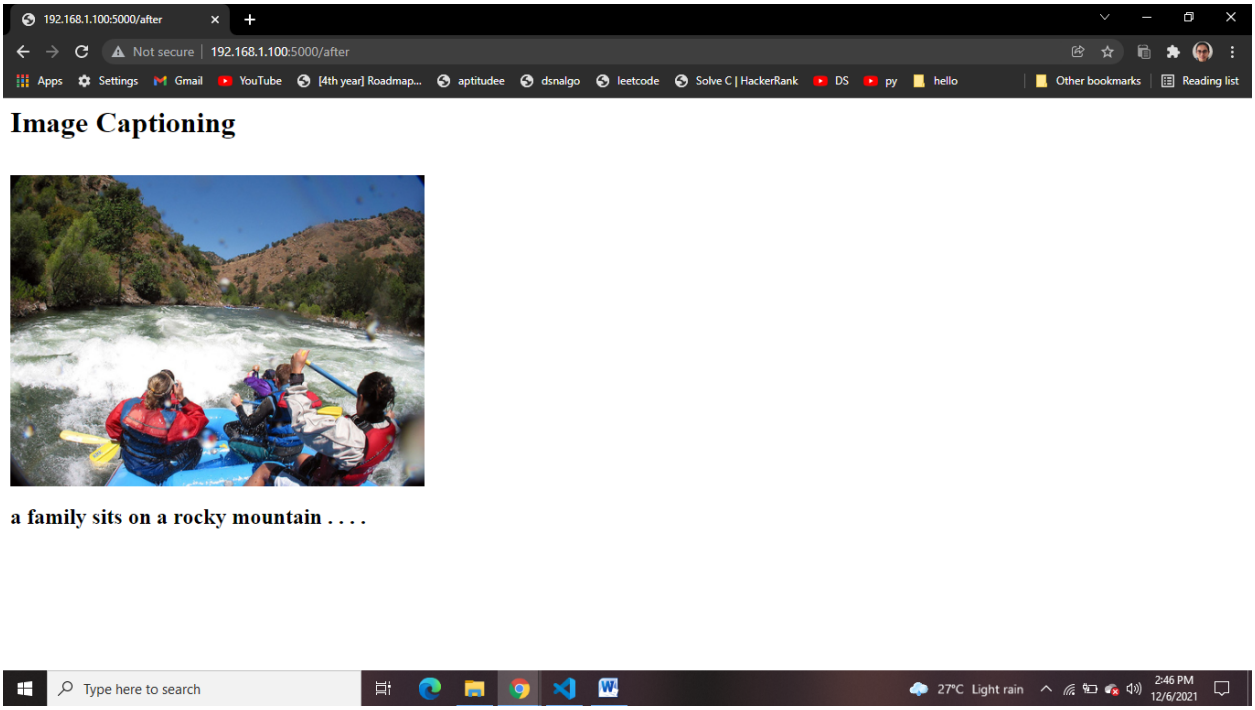

192.168.1.100:5000/after
Not secure
192.168.1.100:5000/after
Apps
Settings
Gmail
YouTube
[4th year] Roadmap...
aptitudee
dsnalgo
leetcode
Solve C | HackerRank
DS
py
hello
Other bookmarks
Reading list
Image Captioning
a hiker wears a snowsuit gear is in front of a mountain . . .
Type here to search
27°C Light rain
1:43 PM
12/6/2021

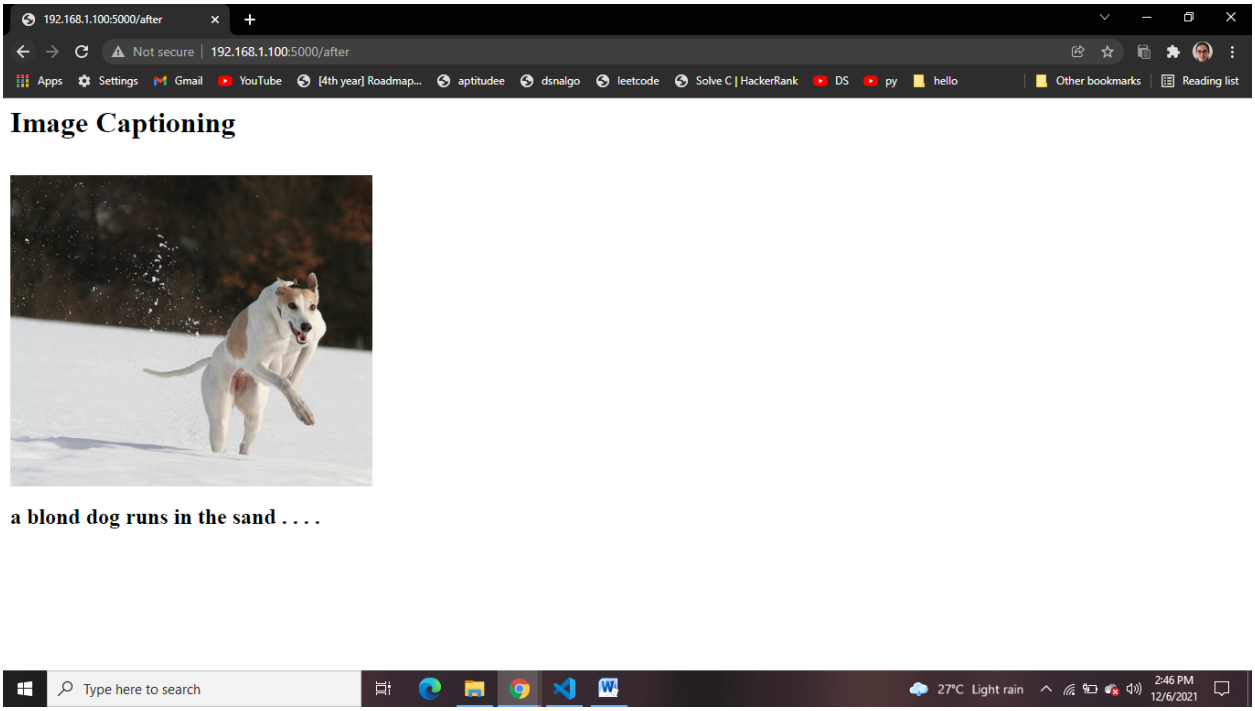

192.168.1.100:5000/after
Not secure | 192.168.1.100:5000/after
Apps Settings Gmail YouTube [4th year] Roadmap... aptitudee dsnalgo leetcode Solve C | HackerRank DS py hello Other bookmarks Reading list
Image Captioning
two dogs play with a small small dog . . .
Type here to search
27°C Light rain
1:46 PM
12/6/2021


192.168.1.100:5000/after
Not secure | 192.168.1.100:5000/after
Apps Settings Gmail YouTube [4th year] Roadmap... aptitudee dsnalgo leetcode Solve C | HackerRank DS py hello Other bookmarks Reading list
Image Captioning
a man in a white uniform jumps while holding a basketball . . .
Type here to search
27°C Light rain
1:47 PM
12/6/2021


192.168.1.100:5000/after
Not secure | 192.168.1.100:5000/after
Apps Settings Gmail YouTube [4th year] Roadmap... aptitudee dsnalgo leetcode Solve C | HackerRank DS py hello Other bookmarks Reading list
Image Captioning
a young boy is playing tennis playing on a ball . . .
Type here to search
27°C Light rain
1:49 PM
12/6/2021


192.168.1.100:5000/after
Not secure | 192.168.1.100:5000/after
Apps Settings Gmail YouTube [4th year] Roadmap... aptitudee dsnalgo leetcode Solve C | HackerRank DS py hello Other bookmarks Reading list
Image Captioning
a woman holding a baby in front of a woman . . .
Type here to search
27°C Light rain
1:50 PM
12/6/2021


192.168.1.100:5000/after
Not secure | 192.168.1.100:5000/after
Apps Settings Gmail YouTube [4th year] Roadmap... aptitudee dsnalgo leetcode Solve C | HackerRank DS py hello Other bookmarks Reading list
Image Captioning
a family sits on a rocky mountain . . . .
Type here to search
27°C Light rain
2:46 PM
12/6/2021


192.168.1.100:5000/after
Not secure | 192.168.1.100:5000/after
Apps Settings Gmail YouTube [4th year] Roadmap... aptitudee dsnalgo leetcode Solve C | HackerRank DS py hello Other bookmarks Reading list
Image Captioning
a blond dog runs in the sand . . . .
Type here to search
27°C Light rain
2:46 PM
12/6/2021

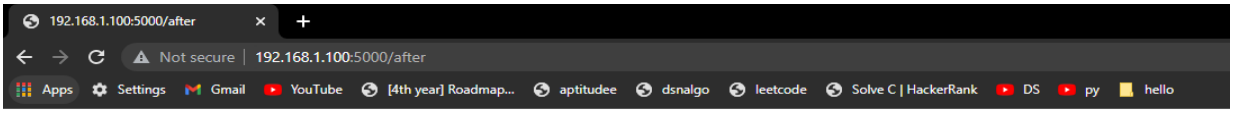


Image Captioning

a person and a person on a sled on a snowy mountain . . . .

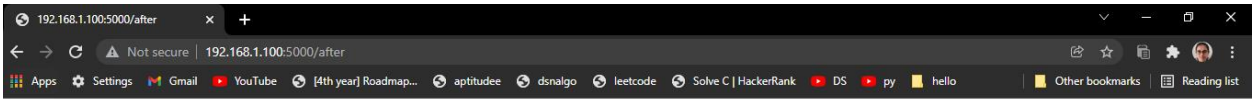


Image Captioning

two men in robes wave at a car wave at an suv in the desert . .

DISCUSSIONS

As you can see the results that some of the results are not that accurate but it is identifying gender of the person and some large objects. If we train the model with large dataset we can improve the results of the model and get better captions. Overall the software will evolve from a computer code to a working product that will see its usefulness on the field. The research work done by the team will reflect on the upcoming product through the modifications in design choices and functionalities. As Flask is used as backend we cant use this approach for big applications.

CHAPTER 9

CONCLUSION AND FUTURE WORK

As we reach the conclusion of the phase 2 of the capstone project we realise that this phase completed with the review of multiple literatures that had been published previously regarding this topic which gave us an insight into the process of Image captioning, and the various algorithms and approaches undertaken by researchers of the same field.This also led us to realise the feasibility of the project and its various implementations or features. Progressing in the project we developed web application of Image Captioning App which generates caption when an image is uploaded into the system. At each stage of the project we enhanced our knowledge by the vast amount of resources present on the internet and the papers to ensure optimum output and compatibility with real life implementations.

FUTURE WORK

- Deployment of localhost Flask Server into any cloud.
- Since the application is big we cant deploy on free or beginner's account in the cloud platforms like heroku,aws,azure,google cloud console.So try deploying on cloud server with upgraded account with maximum resources.
- After deploying we can use the url generated in creating Flutter application which can be used by android/ios users which will be a very helpful application for blind people.

REFERENCE / BIBLIOGRAPHY

[1] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, Yoshua Bengio ; Proceedings of the 32nd International Conference on Machine Learning, PMLR 37:2048-2057, 2015

[2] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-up and top-down attention for image captioning and visual question answering," in CVPR, 2018

[3] O. Vinyals, A. Toshev, S. Bengio and D. Erhan, "Show and Tell: Lessons Learned from the 2015 MSCOCO ImageCaptioning Challenge," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 652-663, 1 April 2017.

[4] R Ani, Effy Maria, J Jameema Joyce, V Sakkaravarthy, MA Raja 2017 International Conference on Innovations in Green Energy and Healthcare Technologies (IGEHT), 1-6, 2017

[5] Doshi, Text Reader for Visually Impaired Using Google Cloud Vision API, international journal of innovative research in technology (IJIRT). Vol. 4, 5/18.

[6] Shuang Liu, Image Captioning Based on Deep Neural Networks, MATEC Web of Conferences 232, 01052 (2018) Available: https://doi.org/10.1051/matecconf/201823201052

[7] Ahmet Aker, generating image descriptions using dependency relational patterns, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 1250–1258, Uppsala, Sweden, 11- 16 July 2010. c 2010 Association for Computational Linguistics.

[8] An Overview of Image Caption Generation Methods, Hindawi Computational Intelligence and Neuroscience Volume 2020, Article ID 3062706, 13 pages https://doi.org/10.1155/2020/3062706

[9] Zhongliang Yang, Yu-Jin Zhang, Sadaqat ur Rehman, Yongfeng Huang, Image Captioning with Object Detection and Localization, [Online] Available: https://arxiv.org/ftp/arxiv/papers/1706/1706.02430.pdf

[10] Image Caption Generator using Big Data and Machine Learning, International Research Journal of Engineering and Technology (IRJET), Vol.7, 4/20.

[11] https://www.kaggle.com/programminghut/imagecaptioning#Visualize-Images-with-captions

Appendix A: Definitions, Acronyms and Abbreviations

1. **Python**: Python is an interpreted, high-level and general-purpose programming language.

2. **Use case Diagram**: It shows the details about the actors involved in the certain use cases of the project in the simple manner in the form of a sample diagram mentioning the actors, use cases , relationships.

3. **API**: Is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other.

4. **ER Diagram** - Entity-Relationship Diagram Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases.

5. **Data Flow Diagram** :A data flow diagram (DFD) maps out the flow of information for any process or system .

6. **Activity Diagram:** An activity diagram is a behavioural diagram i.e. it depicts the behaviour of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed

7. **Novelty:** Our web app is feasible, unambiguous, understandable, clear and correct.

8. **Performance:** It takes in the given image and analyses it.and gives the captions.

9. **Reliability:** The software will meet all of the functional requirements without any unexpected behavior within a time span designed.

10. **Flask:** It is a web framework allows you to develop web applications with easy core.