VISUAL IMAGE CAPTION GENERATOR USING DEEP LEARNING

Sathyabama Institute of Science and Technology

(DEEMED TO BE UNIVERSITY)

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in

Computer Science and Engineering By

SRUTHI RAVI

(Reg. No. 38110562)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY) Accredited with Grade "A" by NAAC I 12B

Status by UGC I Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

March – 2022

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

(Established under Section 3 of UGC Act, 1956)
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600119
www.sathyabamauniversity.ac.in

SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Sruthi Ravi (38110562)** carried out the project entitled "**Visual Image Caption Generator using Deep Learning**" under our supervision from November 2021 to March 2022.

Internal Guide
Dr. Veena K.

Head of the Department
Dr. S. VIGNESHWARI, M.E., Ph.D.,
Dr. LAKSHMANAN L, M.E., Ph.D.

Submitted for Viva voce Examination held on_____

Internal Examiner

Dr.Veena K

External Examiner

Dr.Srinivasulu

MS.C.M.Suja

DECLARATION

I, **SRUTHI.R (Reg. No. 38110562)** hereby declare the Project Report entitled **"Visual Image Caption Generator using Deep Learning"** done by me under the guidance of Dr.Veena K. at SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

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

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

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.VEENA. K, M.E, Ph.D.** for her valuable guidance, suggestions, and constant encouragement paved the way for the successful completion of my project work.

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

ABSTRACT

The combination of computer vision and natural language processing in Artificial intelligence has sparked a lot of interest in research in recent years, thanks to the advent of deep learning. The context of a photograph is automatically described in English. When a picture is captioned, the computer learns to interpret the visual information of the image using one or more phrases. The ability to analyze the state, properties, and relationship between these objects is required for the meaningful description generation process of high-level picture semantics. Using CNN -LSTM architectural models on the captioning of a graphical image, we hope to detect things and inform people via text messages in this research. To correctly identify the items, the input image is first reduced to grayscale and then processed by a Convolution Neural Network (CNN). The COCO Dataset 2017 was used. The proposed method for blind individuals is intended to be expanded to include persons with vision loss to speech messages to help them reach their full potential and to track their intellect. In this project, we follow a variety of important concepts of image captioning and its standard processes, as this work develops a generative CNN-LSTM model that outperforms human baselines

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
NLP	Natural Language Processing
DL	Deep Learning
FC	Fully Connected
LRCN	Long-term Recurrent Convolutional Network

CHAPTER 1

INTRODUCTION

1.1 Introduction

Every day, we are bombarded with photos in our surroundings, on social media, and in the news. Only humans are capable of recognizing photos. We humans can recognize photographs without their assigned captions, but machines require images to be taught first. The encoder-decoder architecture of Image Caption Generator models uses input vectors to generate valid and acceptable captions. This paradigm connects the worlds of natural language processing and computer vision. It's a job of recognizing and evaluating the image's context before describing everything in a natural language like English.

Our approach is based on two basic models: CNN (Convolutional Neural Network) and LSTM (Long Short-Term Memory). CNN is utilized as an encoder in the derived application to extract features from the snapshot or image, and LSTM is used as a decoder to organize the words and generate captions. Image captioning can help with a variety of things, such as assisting the visionless with text-to-speech through real-time input about the scenario over a camera feed, and increasing social medical leisure by restructuring captions for photos in social feeds as well as spoken messages.

Assisting children in recognizing chemicals is a step toward learning the language. Captions for every photograph on the internet can result in faster and more accurate authentic photograph exploration and indexing. Image captioning is used in a variety of sectors, including biology, business, the internet, and in applications such as self-driving cars wherein it could describe the scene around the car, and CCTV cameras where the alarms could be raised if any malicious activity is observed. The main purpose of this research article is to gain a basic understanding of deep learning methodologies.

1.1.2 Objective

1. The project aims to work on one of the ways to context a photograph in simple English sentences using Deep Learning (DL).
2. The need to use CNN and LSTM instead of working with RNN

1.1.3 Scope

Our project extends and is being used in any large-scale business industry and also small-scale business industry.

1.2 Statement Problem

In our world, information is considered valuable and some humans face a serious problem regarding visualizing an image. We hence dig into this matter, considering blindness as a major factor, and generate a sentence by allowing users to upload or scan a visual image.

Advantage

- Recommendations in Editing Applications
- Assistance for visually impaired
- Social Media posts
- Self-Driving cars
- Robotics
- Easy to implement and connect to new data sources

Disadvantages

- Do not make intuitive feature observations on objects or actions in the image
- Nor do they give an end-to-end mature general model to solve this problem

CHAPTER 2

LITERATURE SURVEY

Literature survey is the most important step in the software development process. Before developing the tool, it is necessary to determine the time factor, economy, and company strength. Once these things are satisfied, then the next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool, they programmers need a lot of external support. This support can be obtained from senior programmers, books, or websites. Before building the system, the above considerations are taken into account for developing the proposed system.

The major part of the project development sector considers and fully surveys all the required needs for developing the project. For every project, a Literature survey is the most important sector in the software development process. Before developing the tools and the associated designing it is necessary to determine and survey the time factor, resource requirement, manpower, economy, and company strength.

To improve and tailor the user experience on its products, photos use image classification. Intraclass variation, occlusion, deformation, size variation, perspective variation, and lighting are all frequent issues in computer vision that are represented by the picture classification problem.

Methods that work well for picture classification are likely to work well for other important computer vision tasks like detection, localization, and segmentation as well.

Image captioning is a great illustration of this. Given an image, the image captioning challenge is to generate a sentence description of the image. The picture captioning problem is comparable to the image classification problem in that it expects more detail and has a bigger universe of possibilities. Image classification is used as a black box system in modern picture captioning systems, therefore greater image classification leads to better captioned.

The image captioning problem is intriguing in and of itself because it brings together two significant AI fields: computer vision and natural language processing. An image captioning system demonstrates that it understands both image semantics and natural language.

Once these things are satisfied and fully surveyed, then the next step is to determine about the software specifications in the respective system such as what type of operating system the project would require, and what all the necessary software is needed to proceed with the next step such as developing the tools, and the associated operations.

To construct an image sentence, image classification is a key stage in the object recognition and picture analysis process. The final output of the image categorization phase might be a statement.

To date, a variety of image captioning techniques have been presented. Several studies have been carried out in attempt to determine the best image captioning technique. It's difficult to pick one approach as the finest of them all because the results and accuracy are dependent on a variety of circumstances.

In order to achieve the most accurate results, traditional approaches have been constantly modified as well as new image captioning techniques invented during the previous few decades.

Each caption generator technique has its own set of benefits and drawbacks. The focus of the research today is on combining the desired qualities of various techniques in order to boost efficiency.

Many high-level tasks, such as image classification, object detection, and, more recently, semantic segmentation, have recently been proven to obtain outstanding results using convolutional neural networks with many layers. A two-stage technique is frequently used, especially for semantic segmentation. Convolutional networks are trained in this way to offer good local pixel-wise data for the second stage, which is often a more global graphical analysis model.

We will use Long short-term memory (LSTM), which is a subset of RNNs, to tackle the problem of Vanishing Gradient. The main goal of LSTM is to solve the problem of Vanishing Gradients. The unique feature of LSTM is that it can keep data values for long periods, allowing it to address the vanishing gradient problem.

When compared to applying RNN, the results revealed that using a mixture of LSTM generated better outcomes.

CNNs employ multilayer convolution to accomplish feature engineering and integrate these features internally, unlike traditional image recognition algorithms. It also employs the pooling and fully connected (FC) layers, as well as SoftMax.

CHAPTER 3

AIM AND SCOPE OF PRESENT INVESTIGATION

3.1 Aim

Development of automatically describing an image with more than natural language sentences which leads to faster information transfer.

3.2 Scope

The application of image captioning in deep learning and securing to become common computing power is one of the main factors that led to techniques for analysis of new and diverse digital data spreading information and response toward users for quick understanding of information with just an image.

3.3 System Requirements

To be used efficiently, all computer software needs certain hardware components or other software resourced to be present on a computer. These prerequisites are known as computer system requirements and are often used as guidelines as opposed to absolute rules. Most software defines two sets of system requirements. Minimum and recommended. With the increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to exist computer systems than technological advancements.

CHAPTER 4

PROJECT IMPLEMENTATION, ALGORITHMS, AND METHODOLOGY

4.1 Introduction

This project is loaded with CNN and LSTM which act as the platform to generate the sentences from a simple image. This can be worked on all applications.

4.2 Hardware Requirements

- System: i3 Processor
- Hard Disk: 500 GB.
- Monitor: 15”LED
- Input Devices: Keyboard, Mouse
- Ram: 4GB.

4.3 Software Requirements

- Platform: Google Colab
- Coding Language: Python

4.4 Working Explanation

1. A user uploads an image that they want to generate a caption for.
2. A gray-scale image is processed through CNN to identify the objects.
3. A gray-scale image is processed through CNN to identify the objects.
4. CNN scans images left-right, and top-bottom, and extracts important image features.
5. By applying various layers like Convolutional, Pooling, Fully Connected, and thus using activation function, we successfully extracted features of every image.
6. It is then converted to LSTM.
7. Using the LSTM layer, we try to predict what the next word could be.

8. Then the application proceeds to generate a sentence describing the image

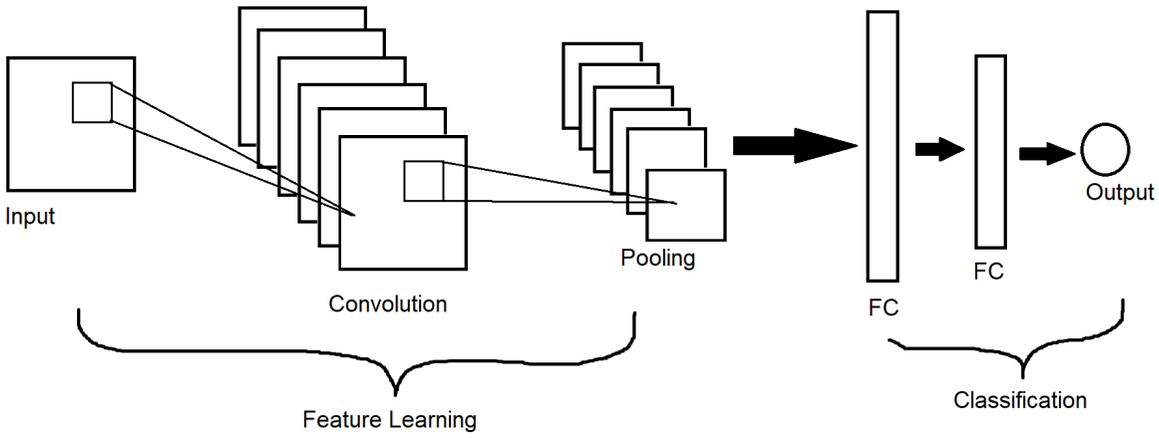
4.5 Algorithms

- Convolutional Neural Network
- Long Short-Term Memory

4.6 Overview on CNN

Convolutional Neural Network (CNN) is a type of deep learning model for processing data that has a grid pattern, such as images.

- deep-learning CNN models to train and test, each input image will pass through a series of convolution layers with filters (Kernals), Pooling, fully connected layers (FC), and apply Softmax function to classify an object with probabilistic values between 0 and 1.
- CNN's have unique layers called convolutional layers which separate them from RNNs and other neural networks.
- Within a convolutional layer, the input is transformed before being passed to the next layer. A CNN transforms the data by using filters.



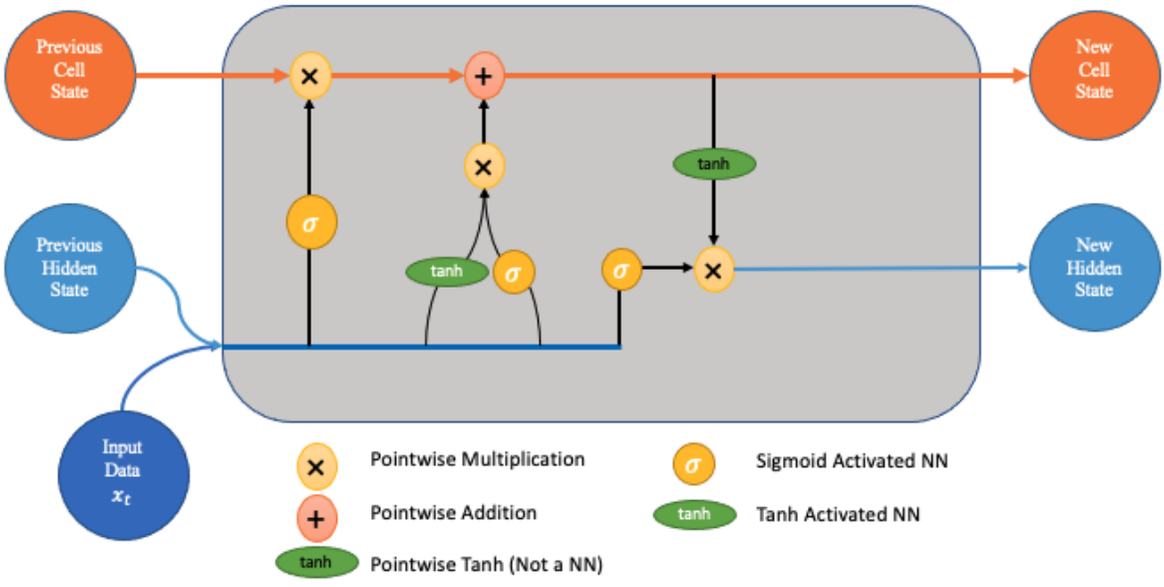
4.6.1 CNN

Some advantages of CNN are:

- It works well for both supervised and unsupervised learning.
- Easy to understand and fast to implement.
- It has the highest accuracy among all algorithms that predicts images.
- Little dependence on pre-processing, decreasing the need for human effort to develop its functionalities.

4.7 Overview of LSTM

LSTM networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems This is a behavior required in complex problem domains like machine translation, speech recognition, and more.
LSTMs are a complex area of deep learning. This is a behavior required in complex problem domains like machine translation, speech recognition, and more. LSTMs are a complex area of deep learning



4.7.1 LSTM

Some advantages of LSTM are:

- Provides us with a large range of parameters such as learning rates, and input and output biases.
- The complexity to update each weight is reduced to O (1) with LSTMs.

4.8 CNN - LSTM Architecture Model

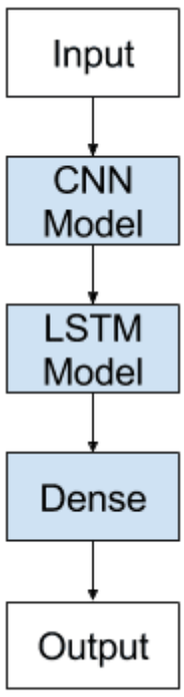
The CNN LSTM architecture involves using Convolutional Neural Network (CNN) layers for feature extraction on input data combined with LSTMs to support sequence prediction.

CNN-LSTMs were developed for visual time series prediction problems and the application of generating textual descriptions from sequence of image (e.g., videos) Specifically, the problem of

- Activity Recognition: Generating a textual description of activity demonstrated in a sequence of images.
- Image Description: Generating a textual description of a single image.
- Video Description: Generating a textual description of a sequence of images.

This architecture was originally referred to as a Long-term Recurrent Convolutional Network (LRCN) model, although we will use the more generic name “CNN LSTM”

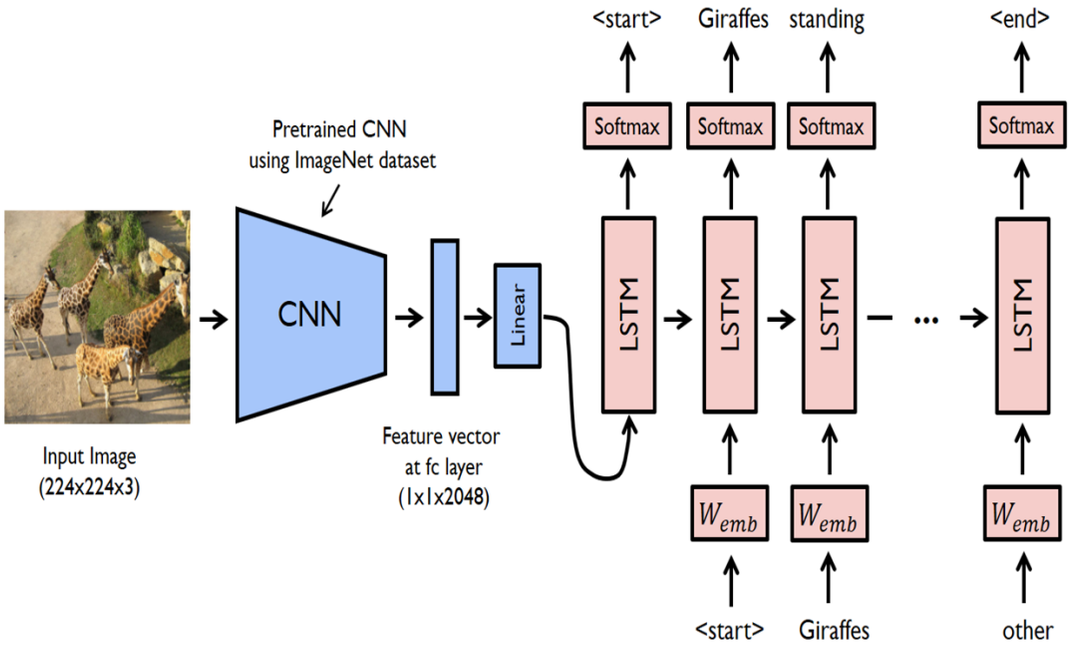
- CNN is used for extracting features from the image. We will use the pre-trained model Xception.
- LSTM will use the information from CNN to help generate a description of the image.



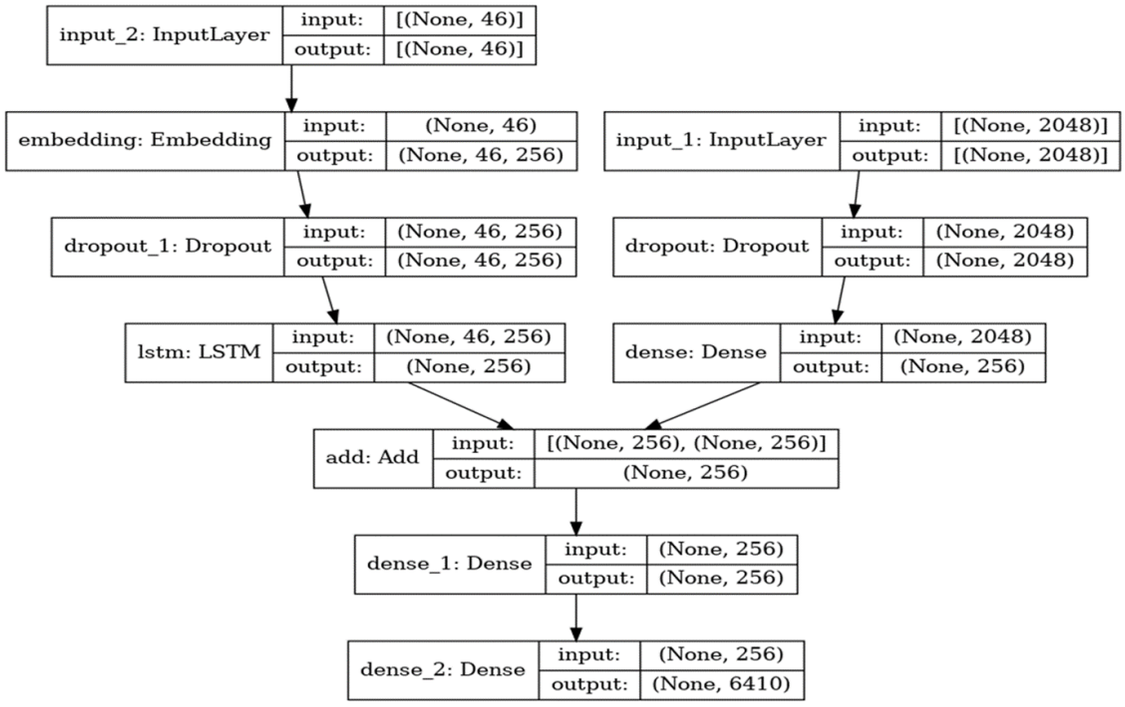
4.8.1 CNN-LSTM model

4.9 Methodology

- Import Libraries
- Upload COCO (Common Objects and Contexts) Dataset 2017. (Data Preprocessing)
- Apply CNN to identify the objects in the image.
- Preprocess and tokenize the captions.
- Use LSTM to predict the next word of the sentence.
- Make a Data Generator
- View Images with caption.



4.9.1 System Architecture



4.9.2 Workflow Diagram

CHAPTER 5

RESULTS AND DISCUSSION

5.1 Results

The result of this program is going to be a user being allowed to generate a caption for a visual image using Deep Learning, NLP, and Computer Vision.

5.1.1 Representation of What Image Captioning Is

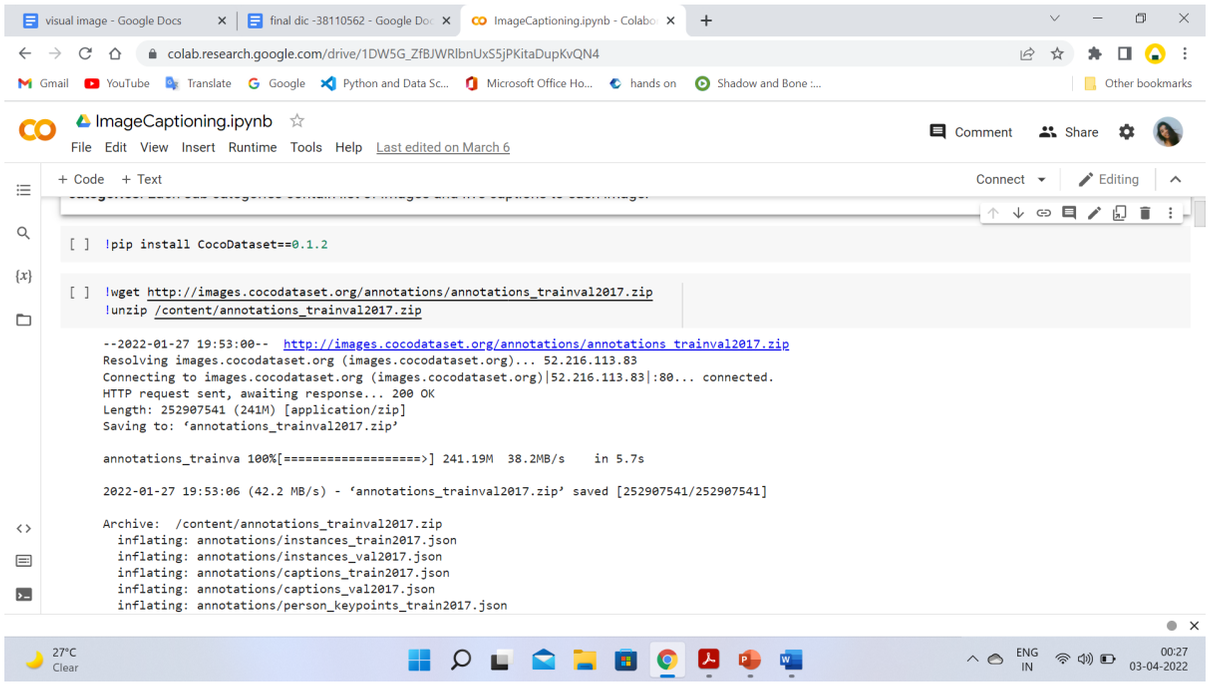
5.2 Discussion

The Dataset that we used for our research is called "COCO Dataset 2017" and is available online. The data was pre-processed to make it suitable for future analysis and work. It consists of 12 main sorts of categories, each with 80 potential sub-categories.

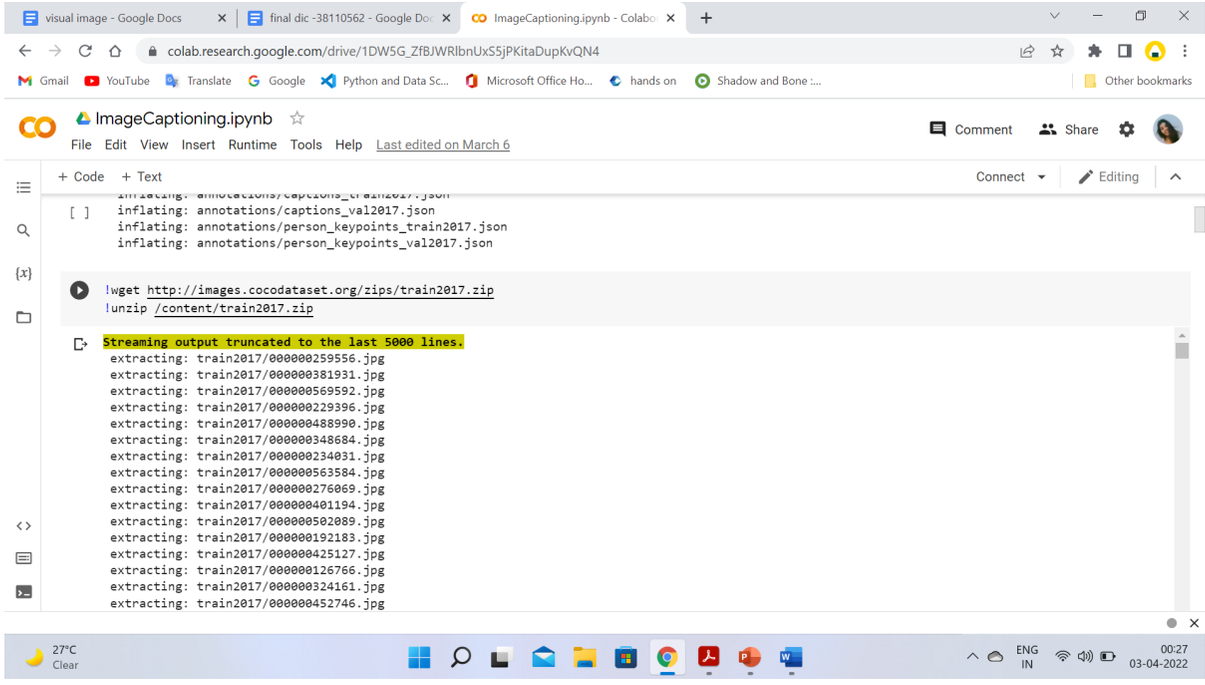
Each subcategory has a collection of photographs as well as five captions for each one. The system performance was evaluated using the general confusion matrix. All of the models' results are listed here, along with their projections. Over the course of 30 iterations, a total of 130 iterations were completed.

CHAPTER 6

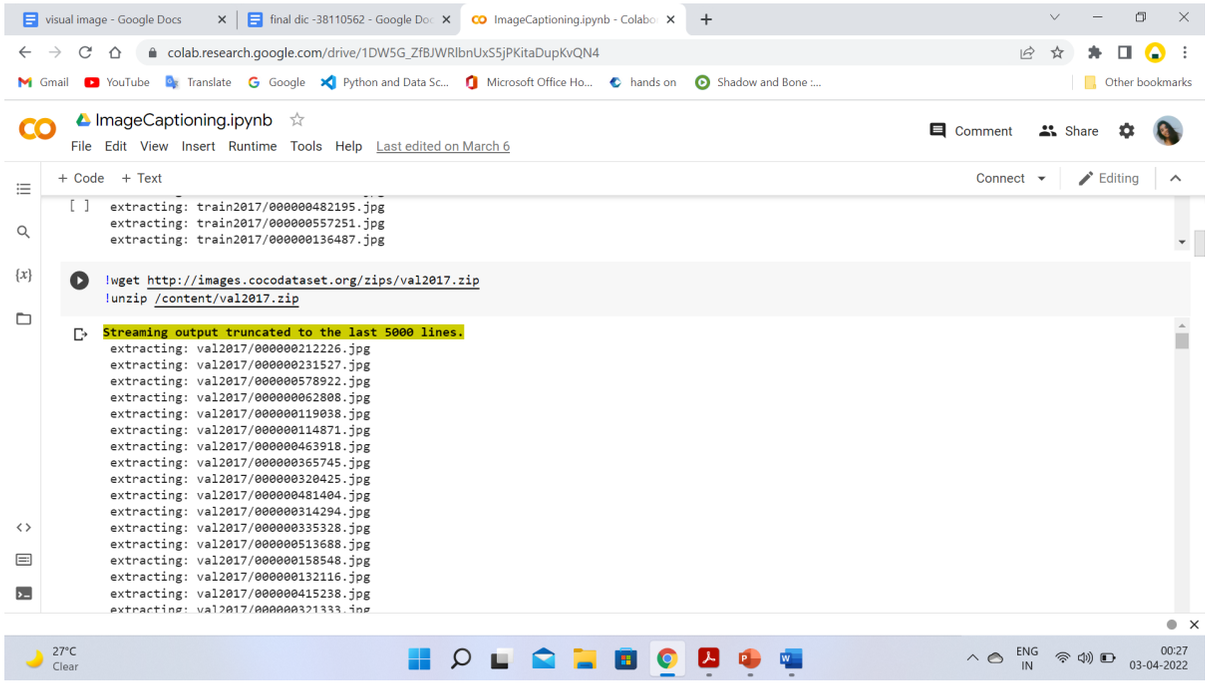
IMPLEMENTED SCREENSHOTS



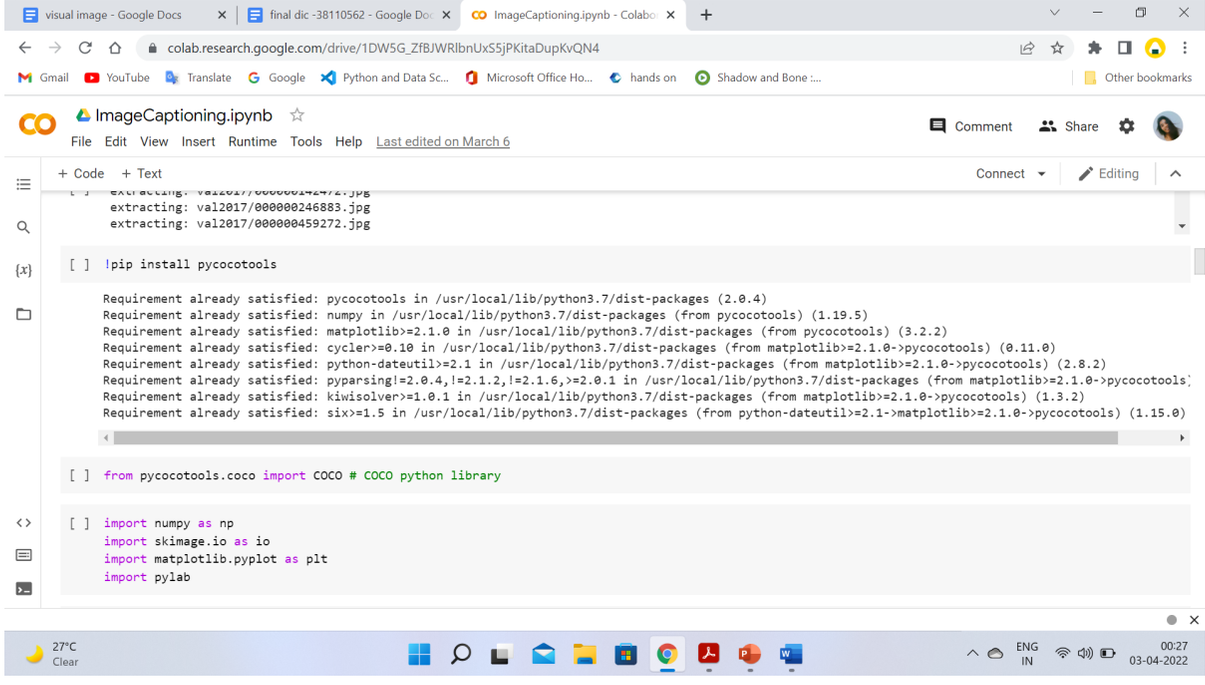
6.1 Pip install



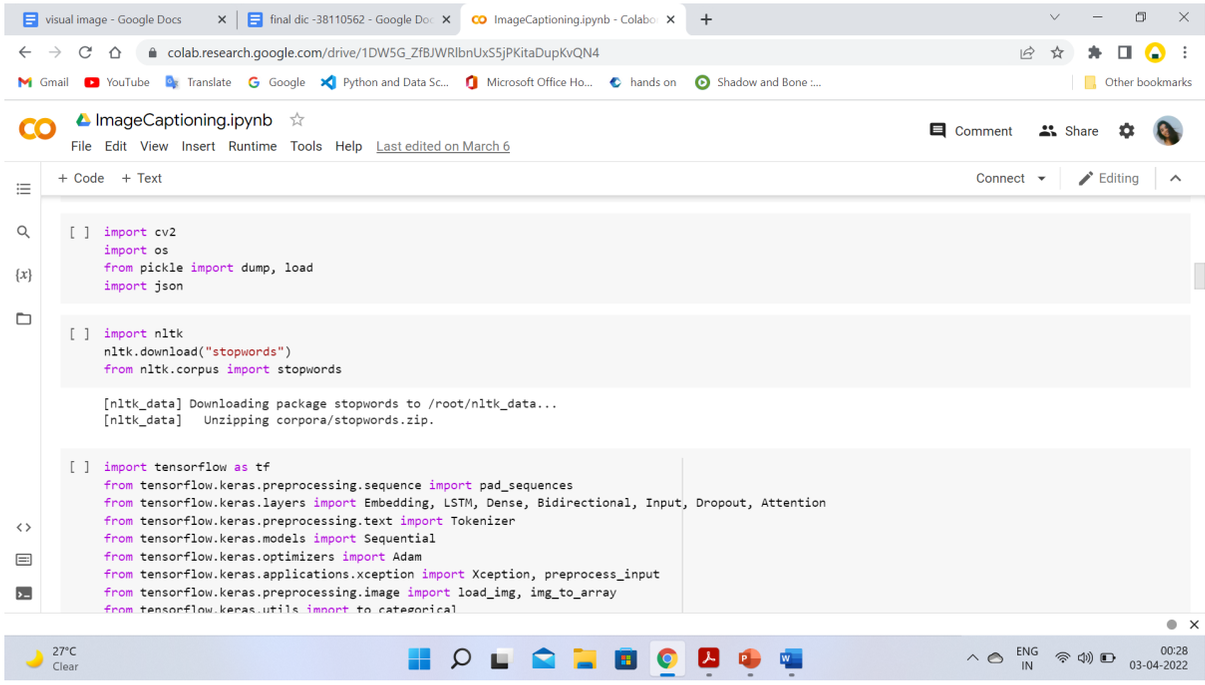
6.2 Load dataset1 & unzip



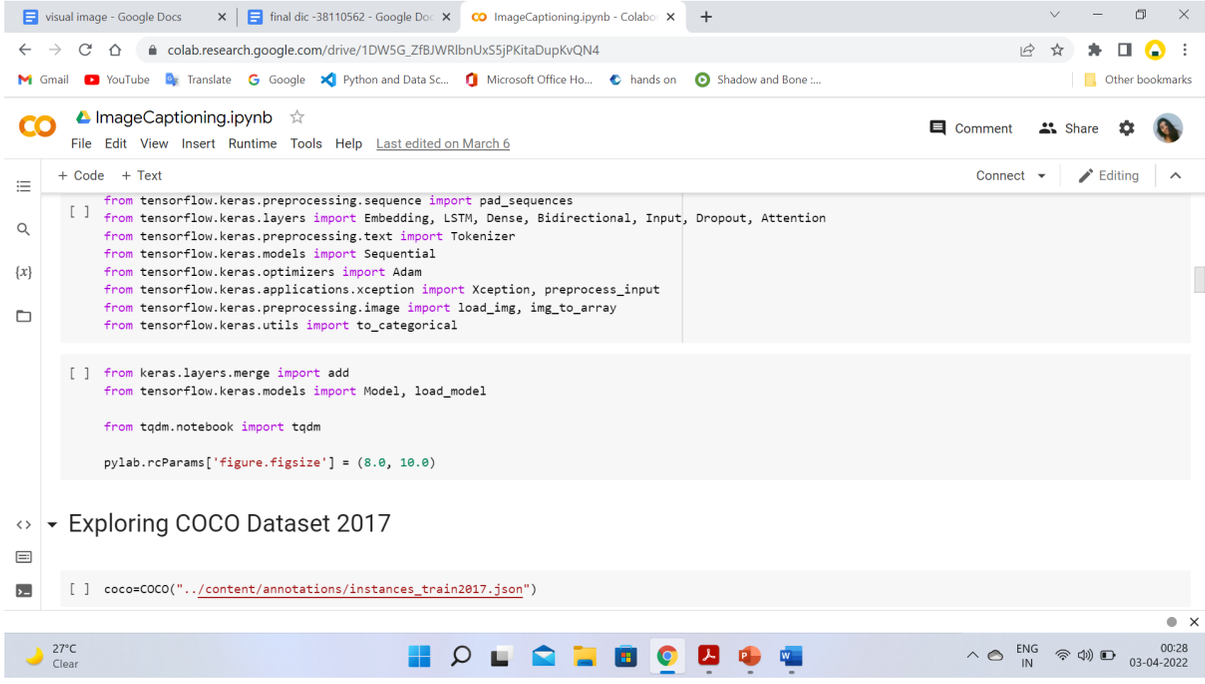
6.3 Load dataset 2 & unzip



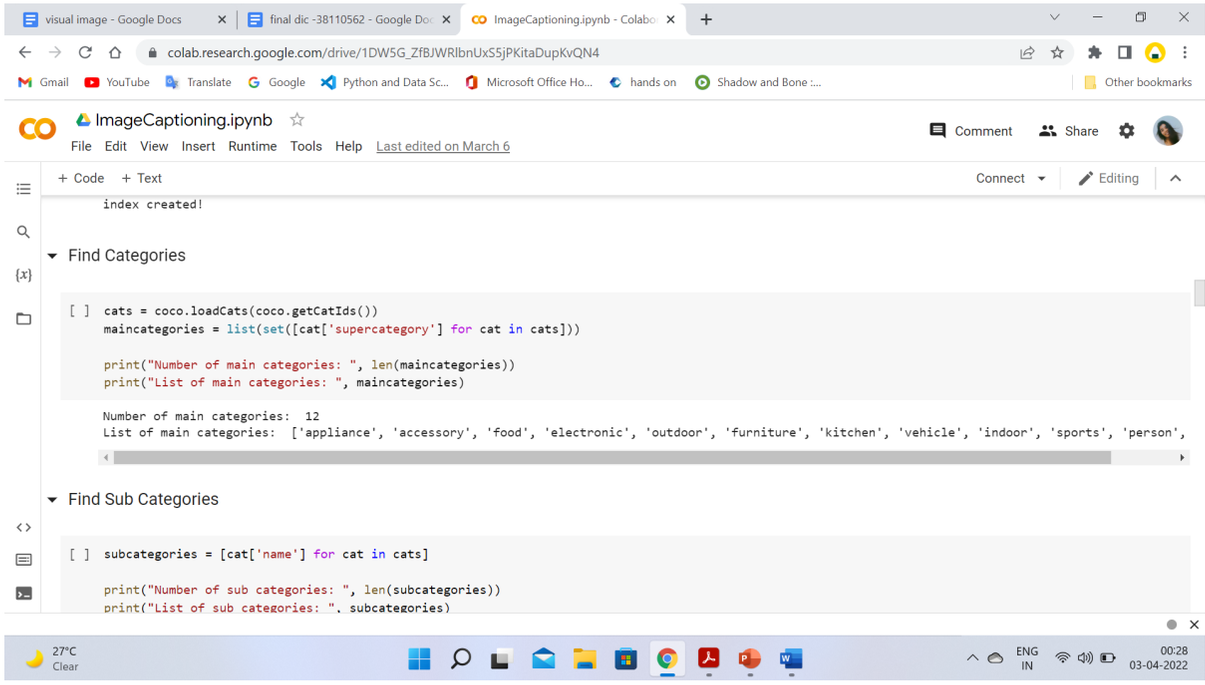
6.4 Install libraires



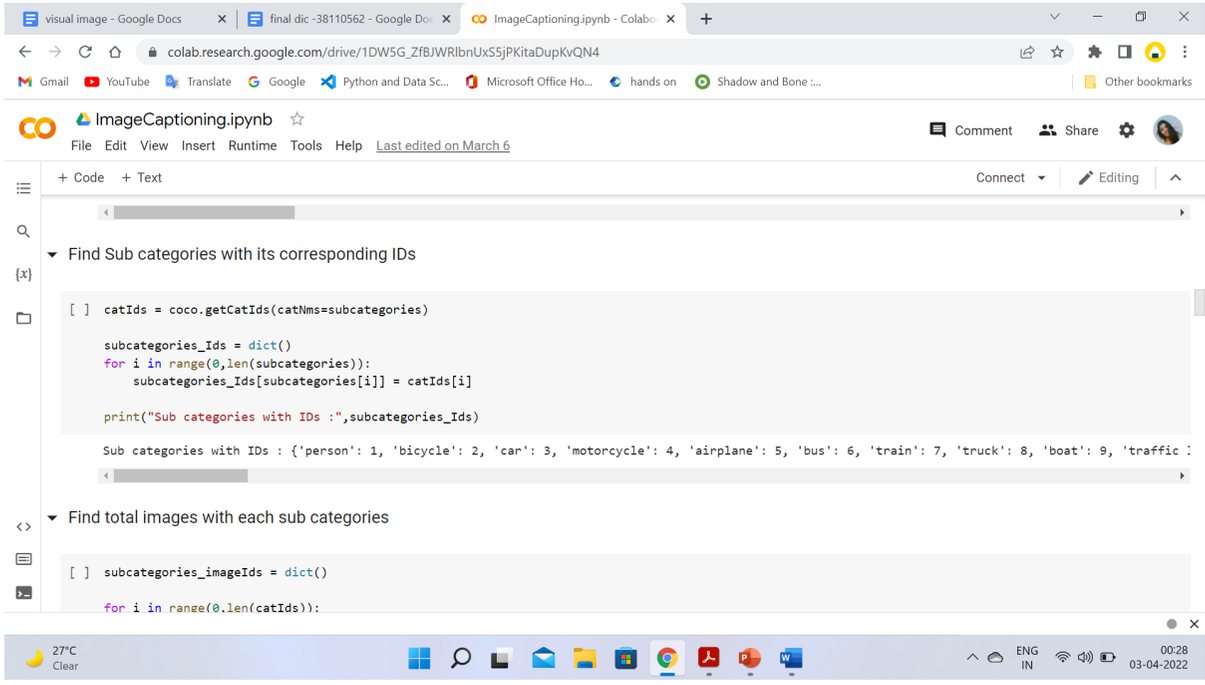
6.5 Import Libraries



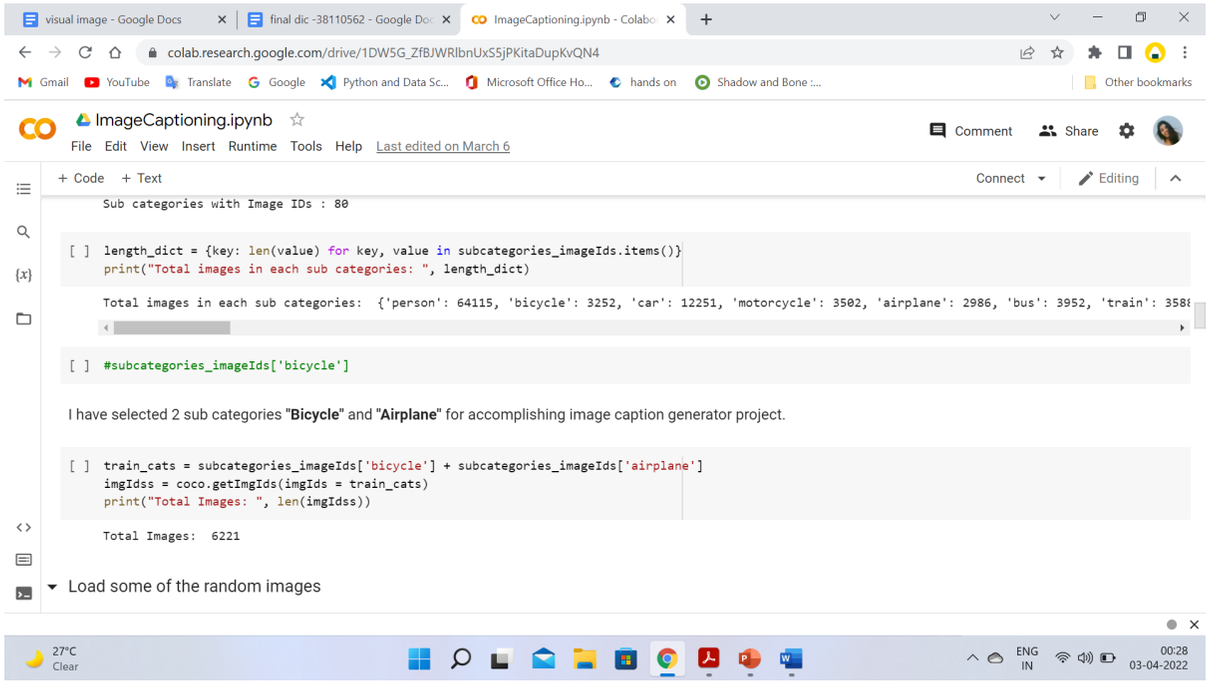
6.6 Import Libraries



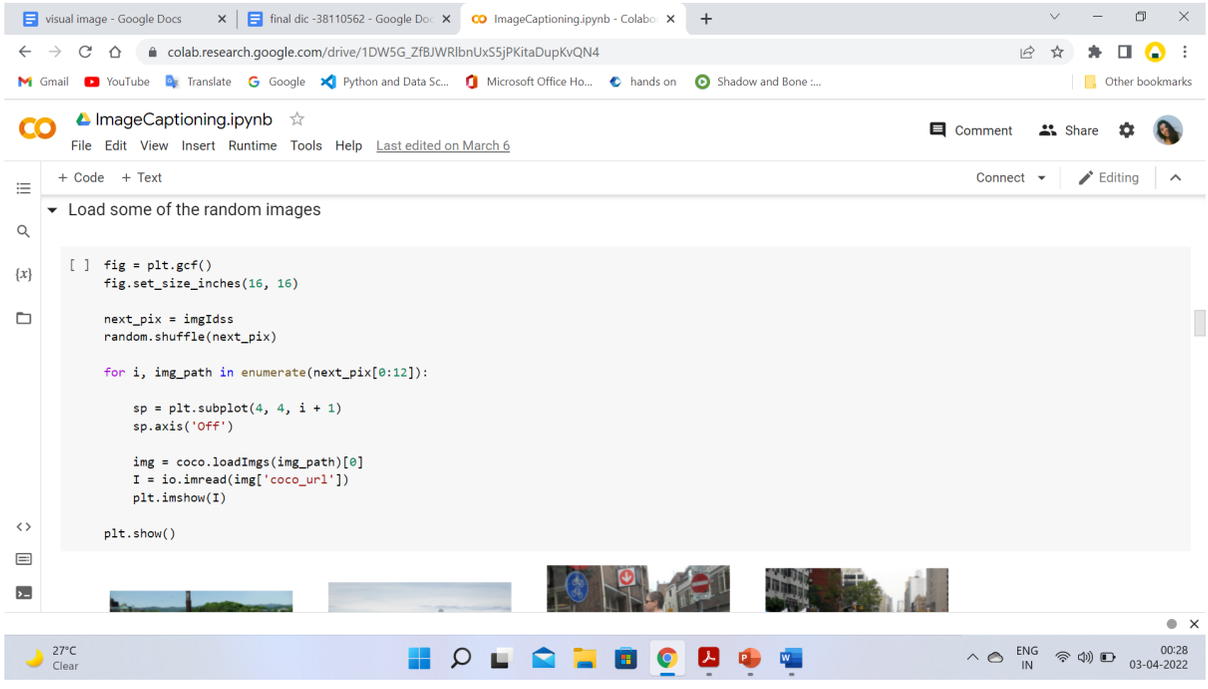
6.7 Find Categories



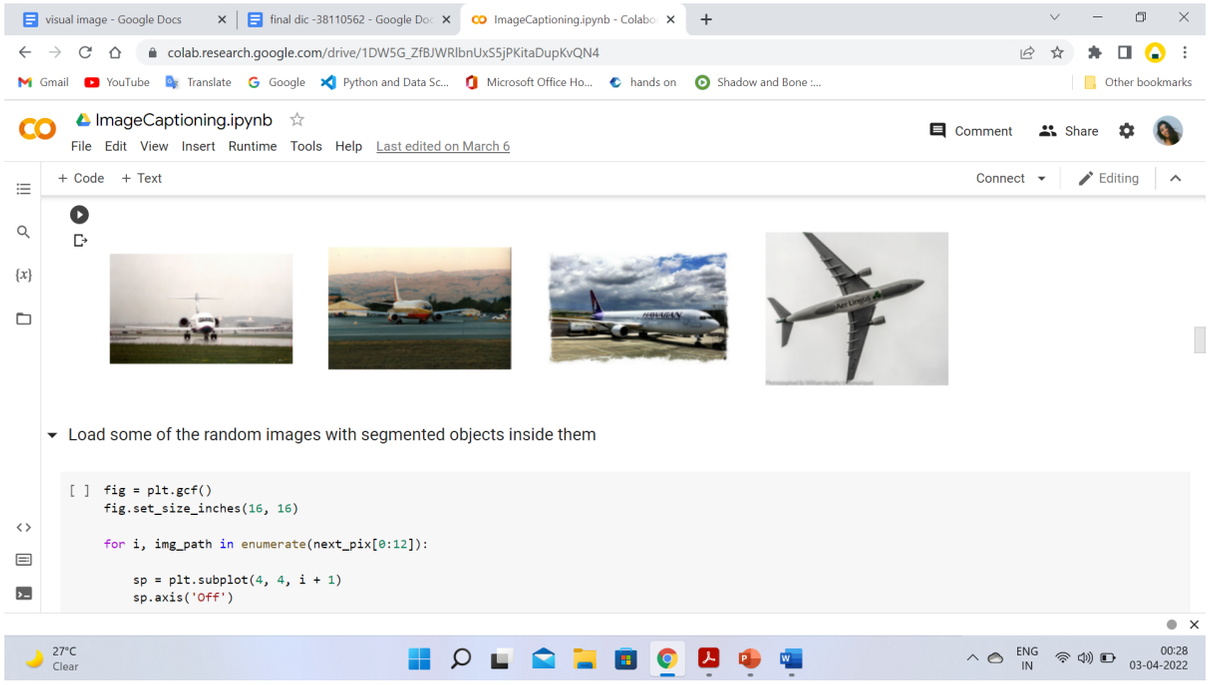
6.8 Find Sub-Categories



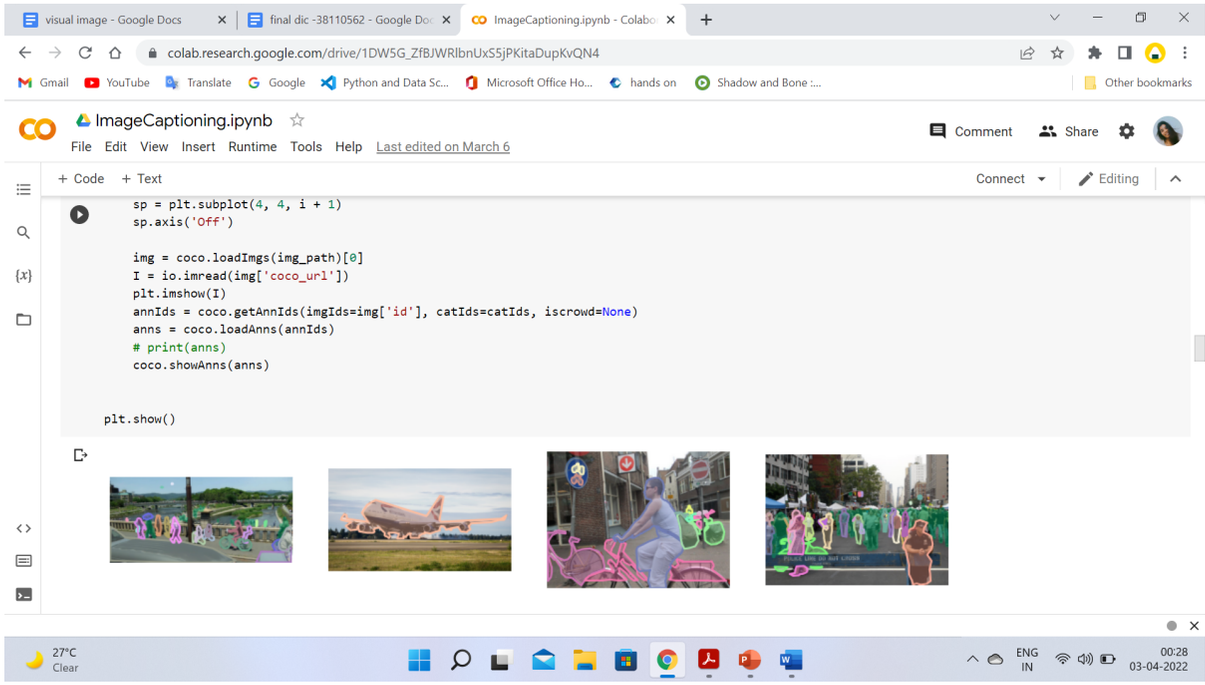
6.9 Find no of images in sub-categories



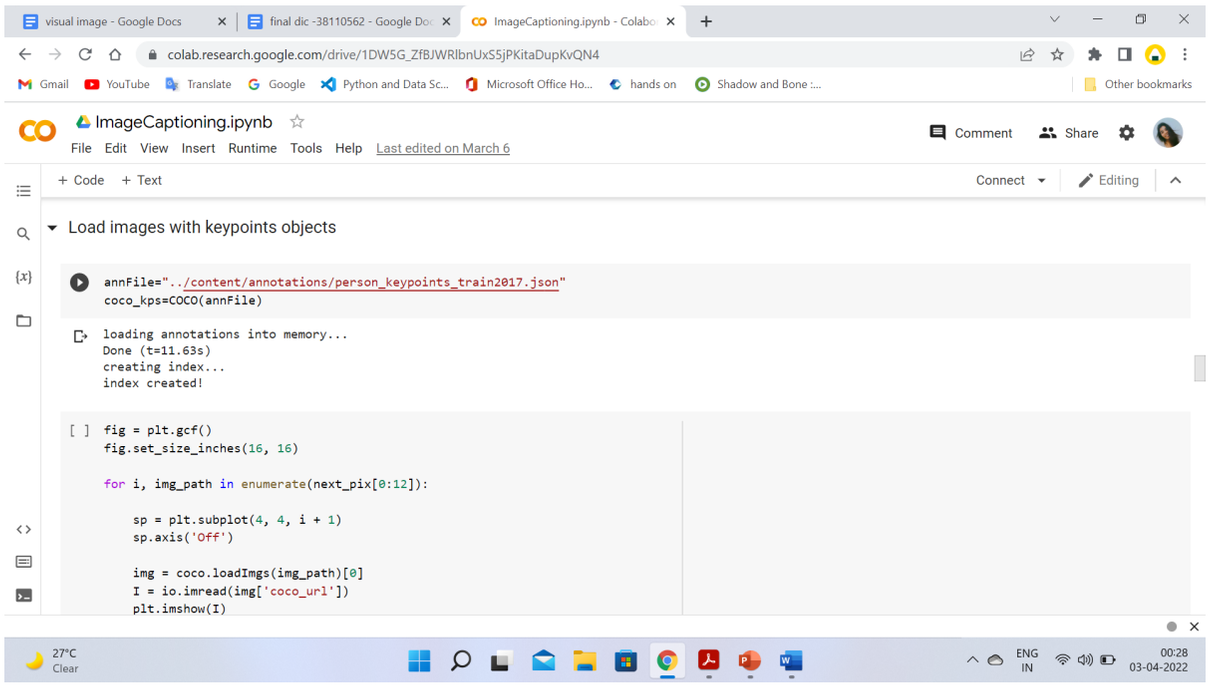
6.10 Load random images



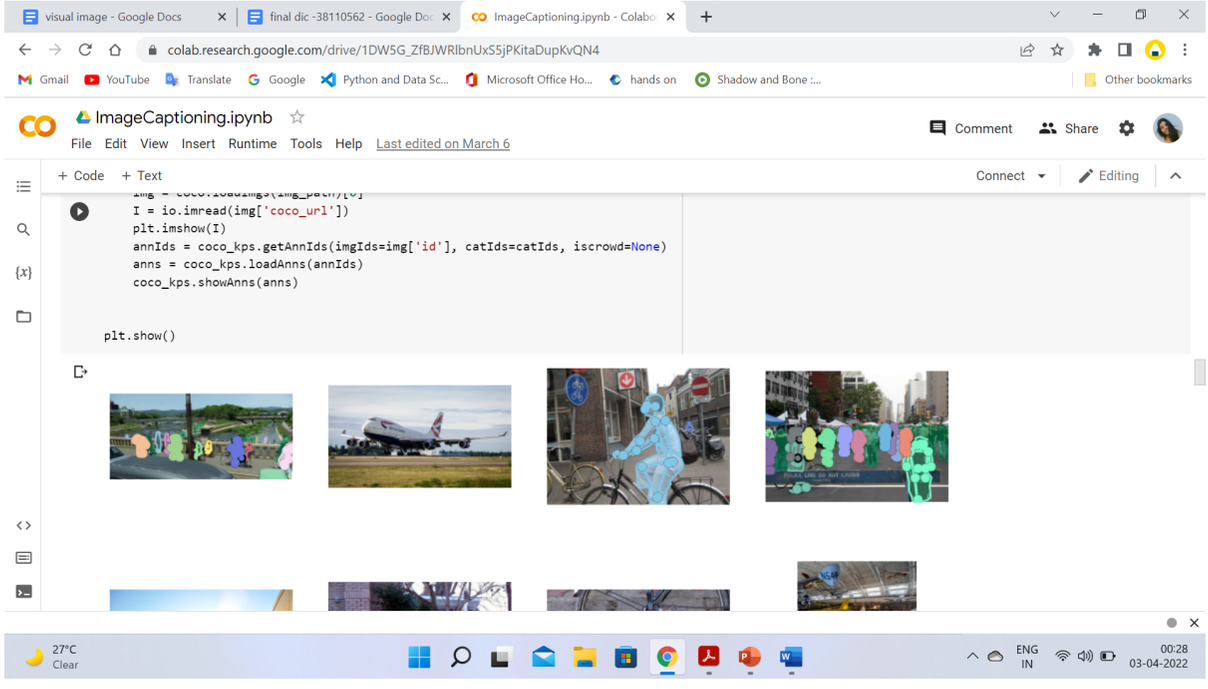
6.11 Load images with segmented objects



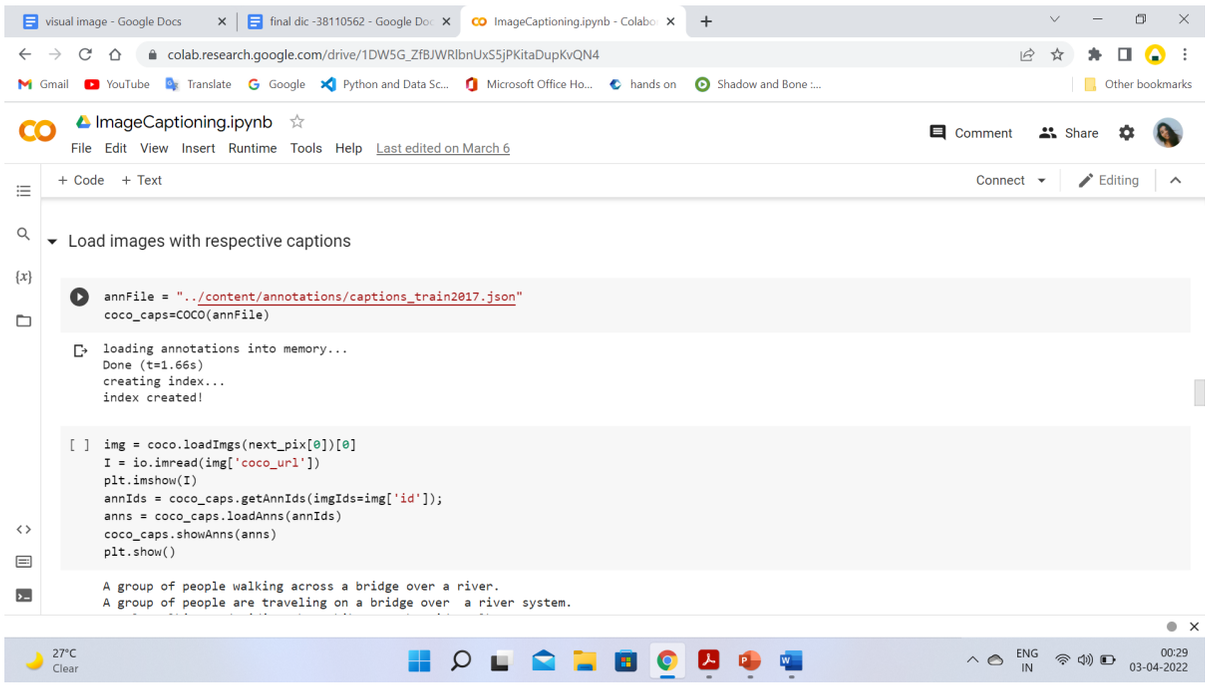
6.12 Load images with segmented objects inside them



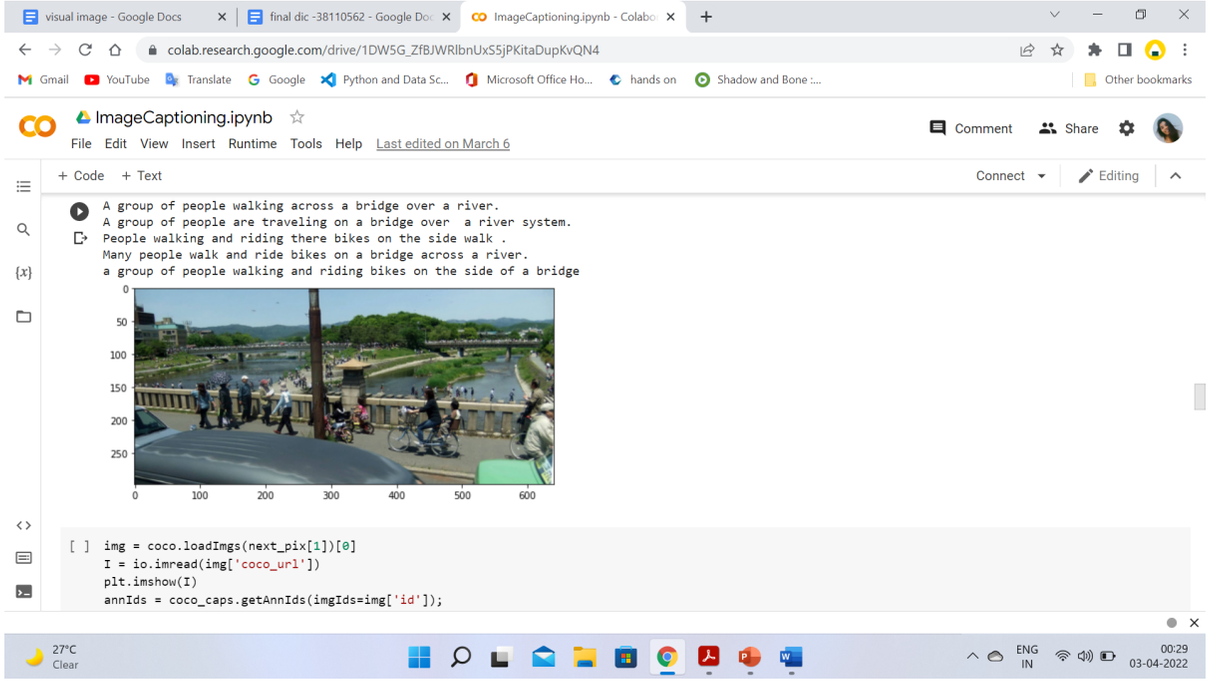
6.13 Load images with keypoints objects



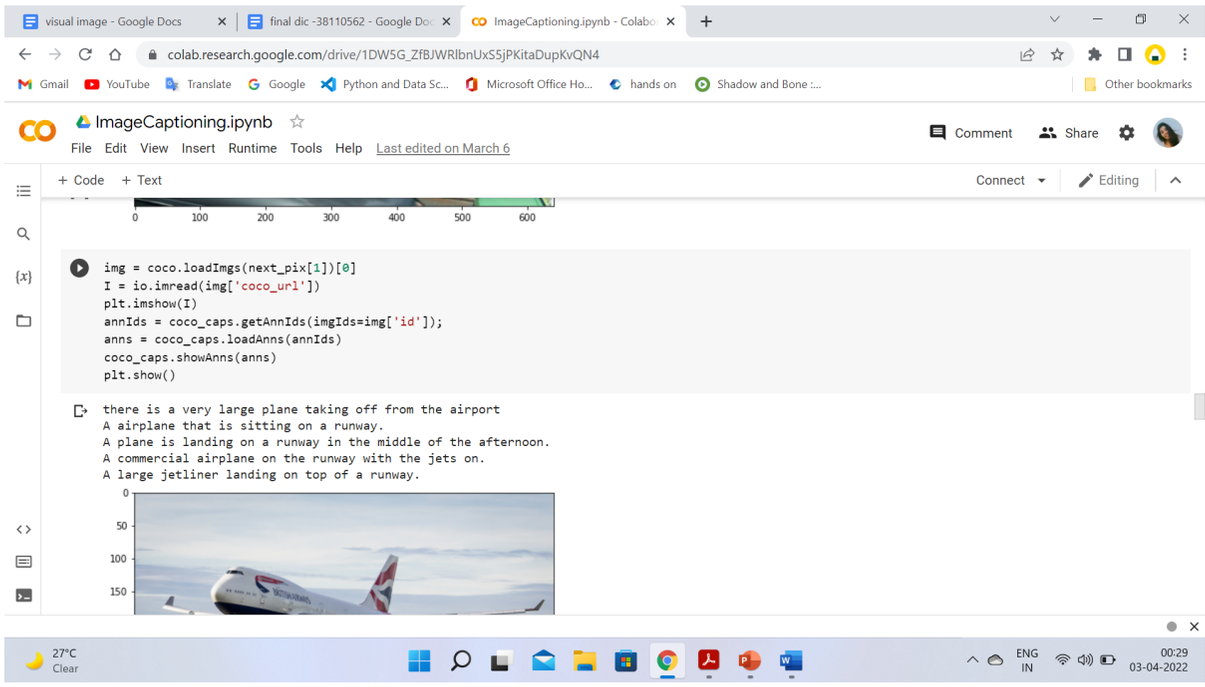
6.14 Result for “Load images with keypoints objects”



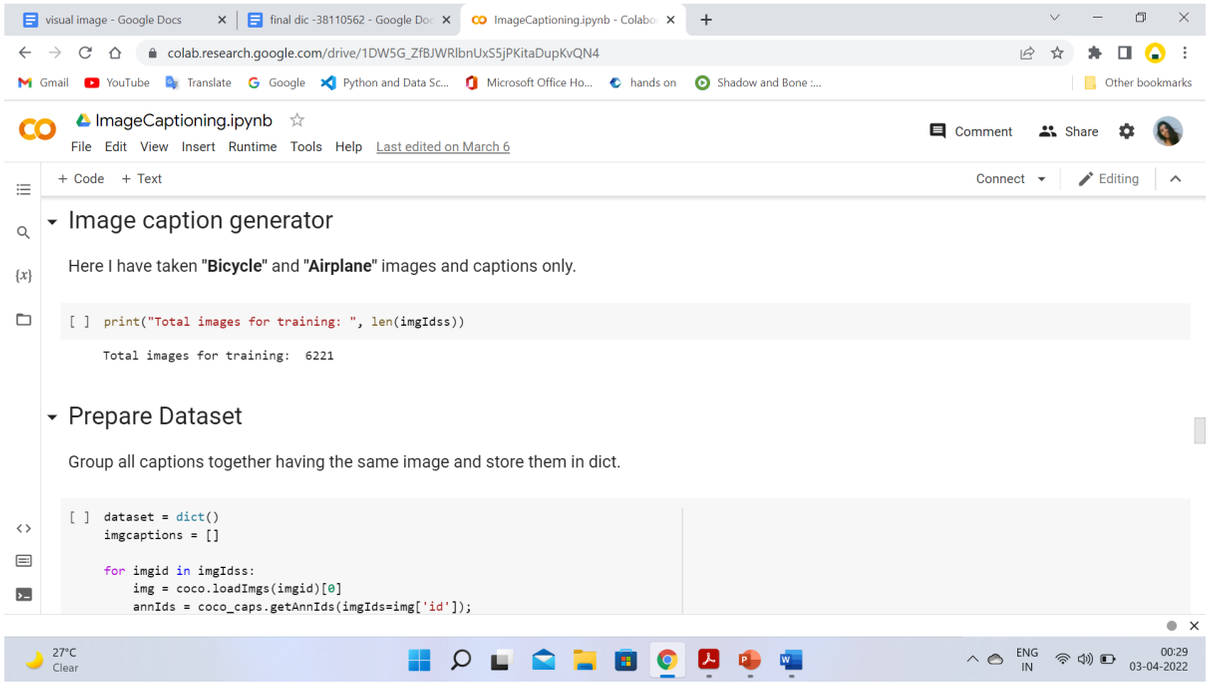
6.15 Load images with respective captions



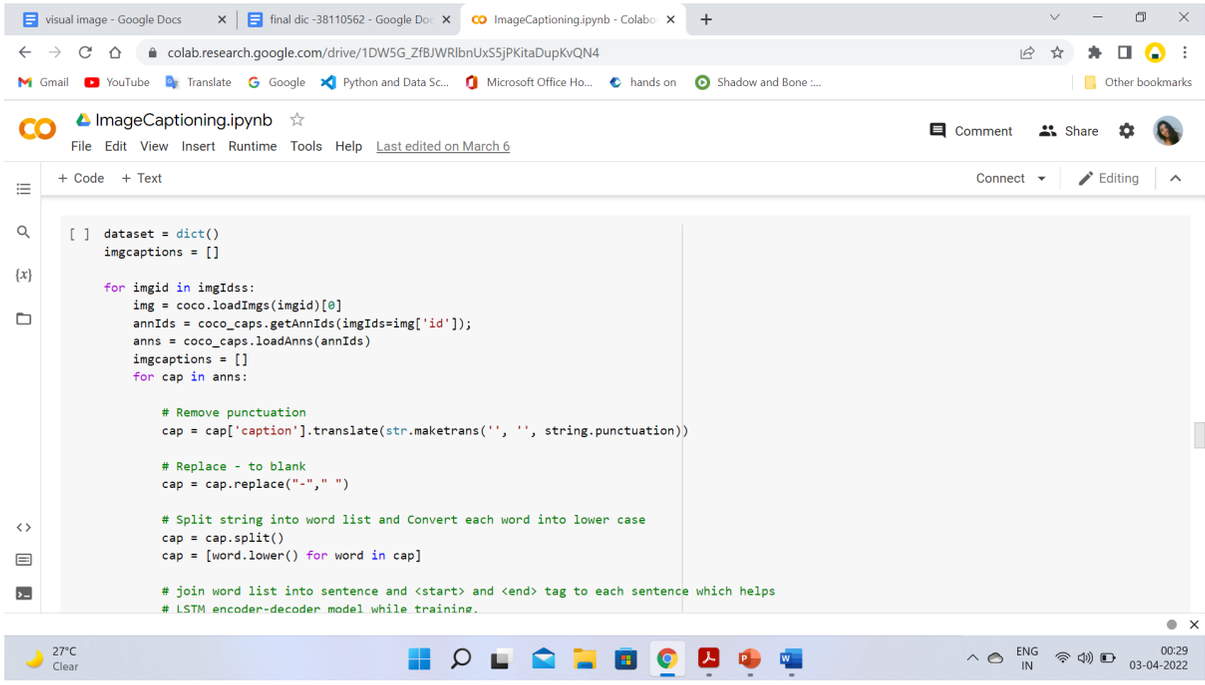
6.16 Output 1 for “Load images with respective captions”



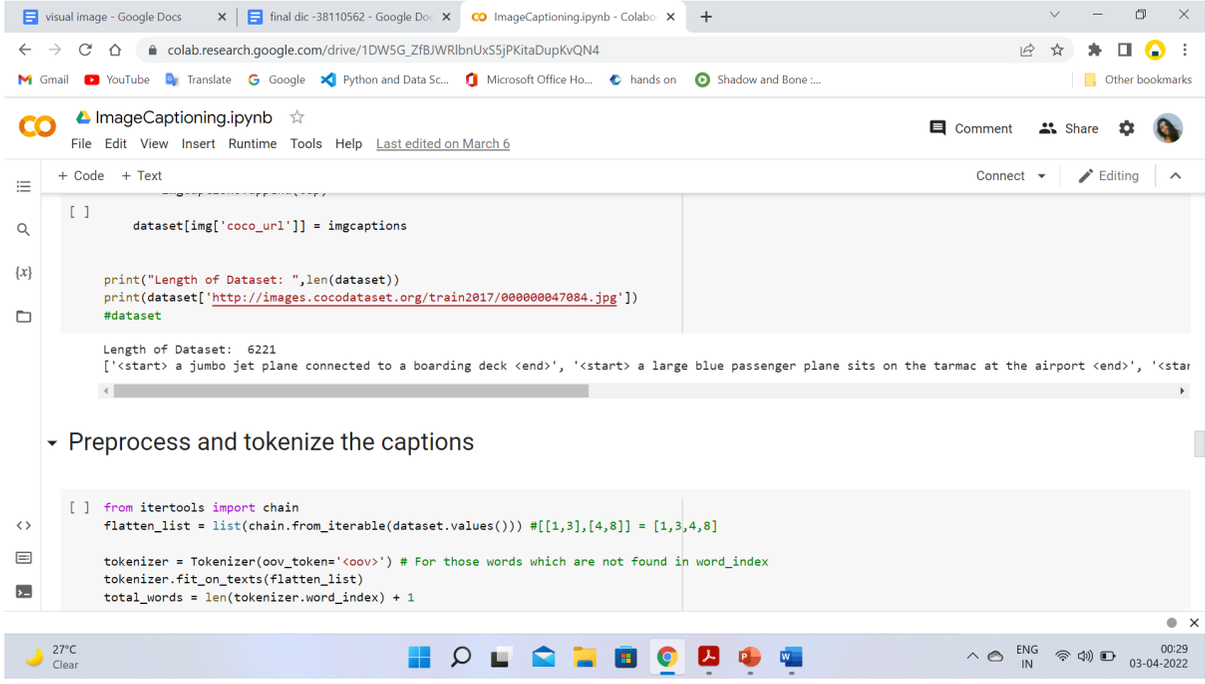
6.17 Output 2 for “Load images with respective captions”



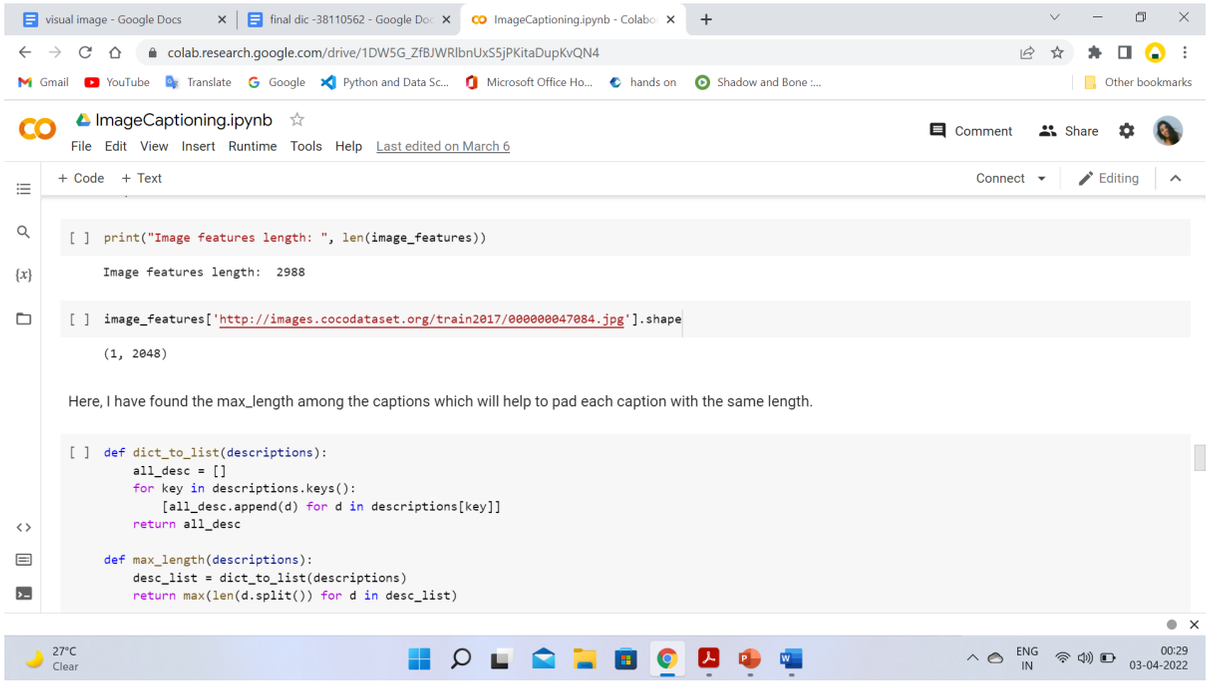
6.18 Prepare dataset



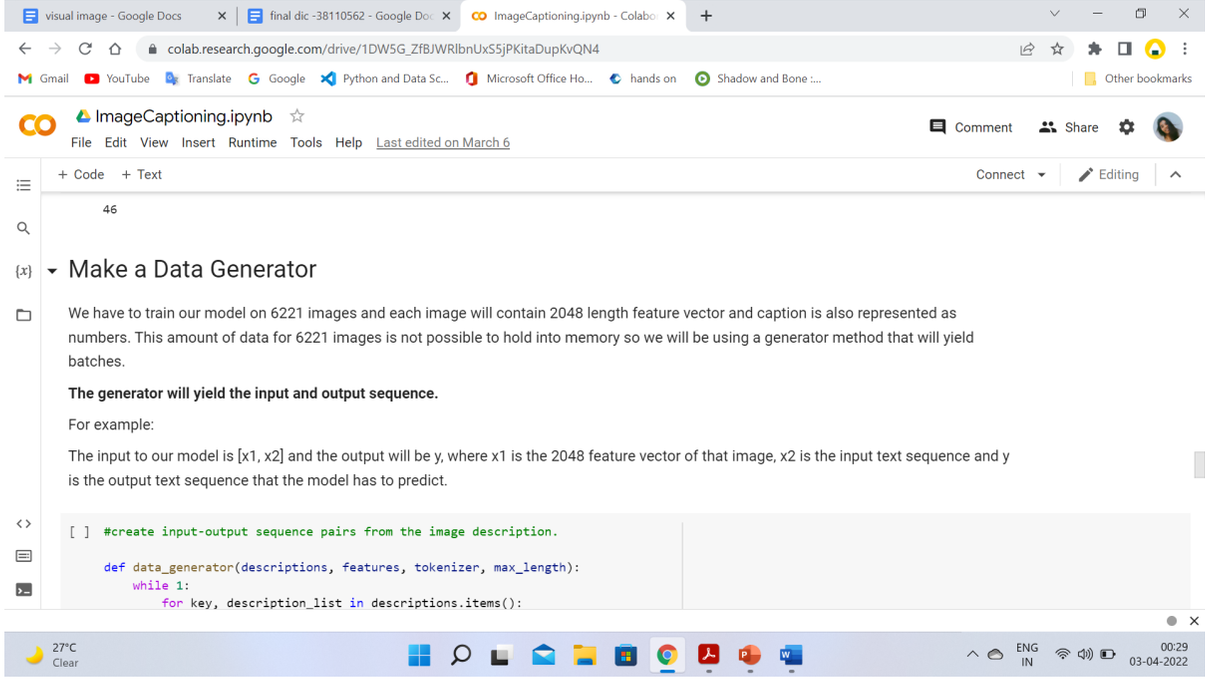
6.19 Code to prepare dataset



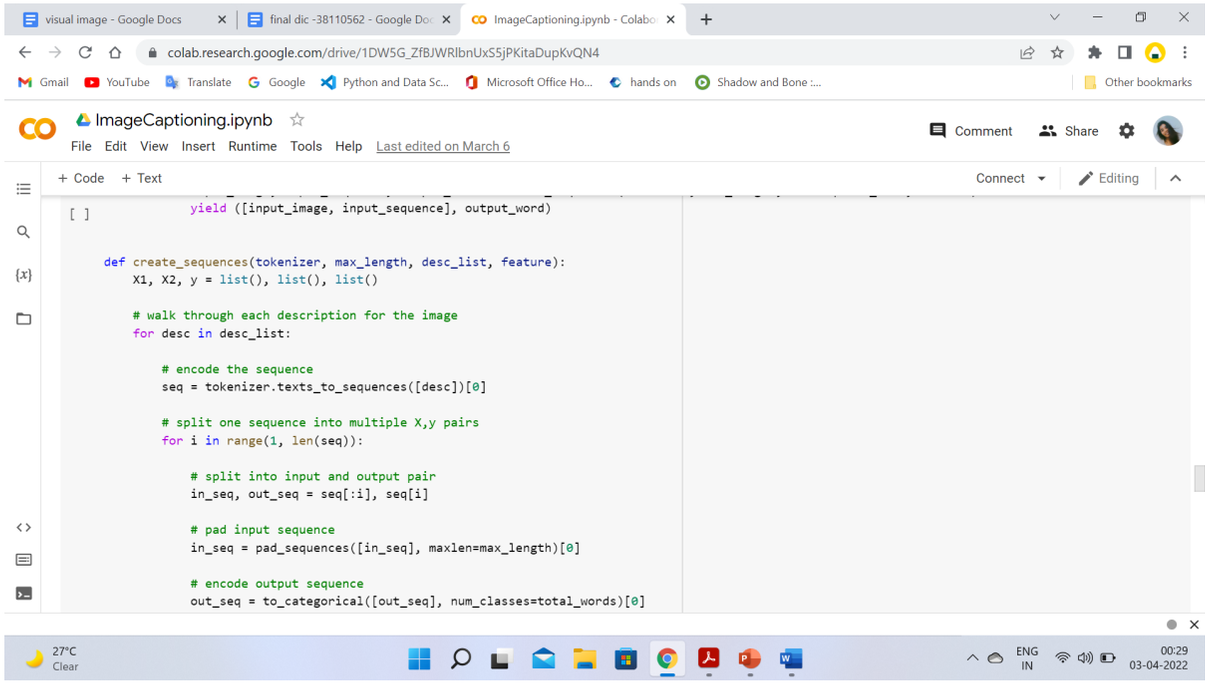
6.20 Preprocess and tokenize the captions



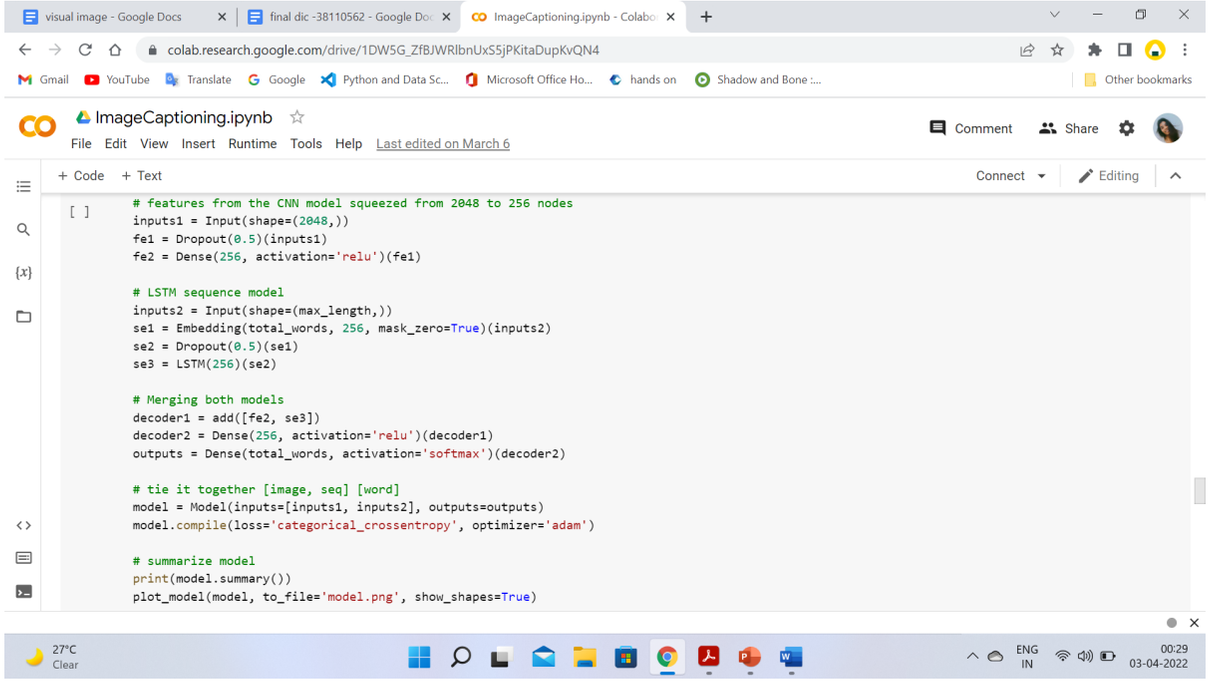
6.21 Preprocess and tokenize the captions



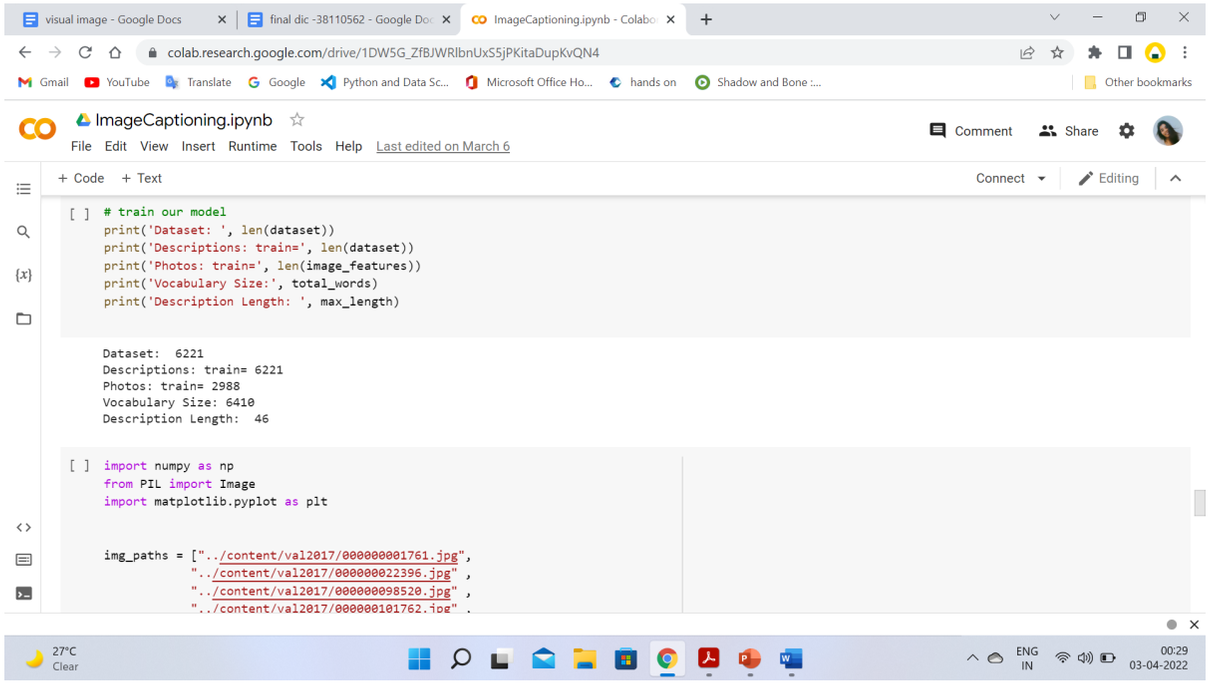
6.22 Explaining data generator



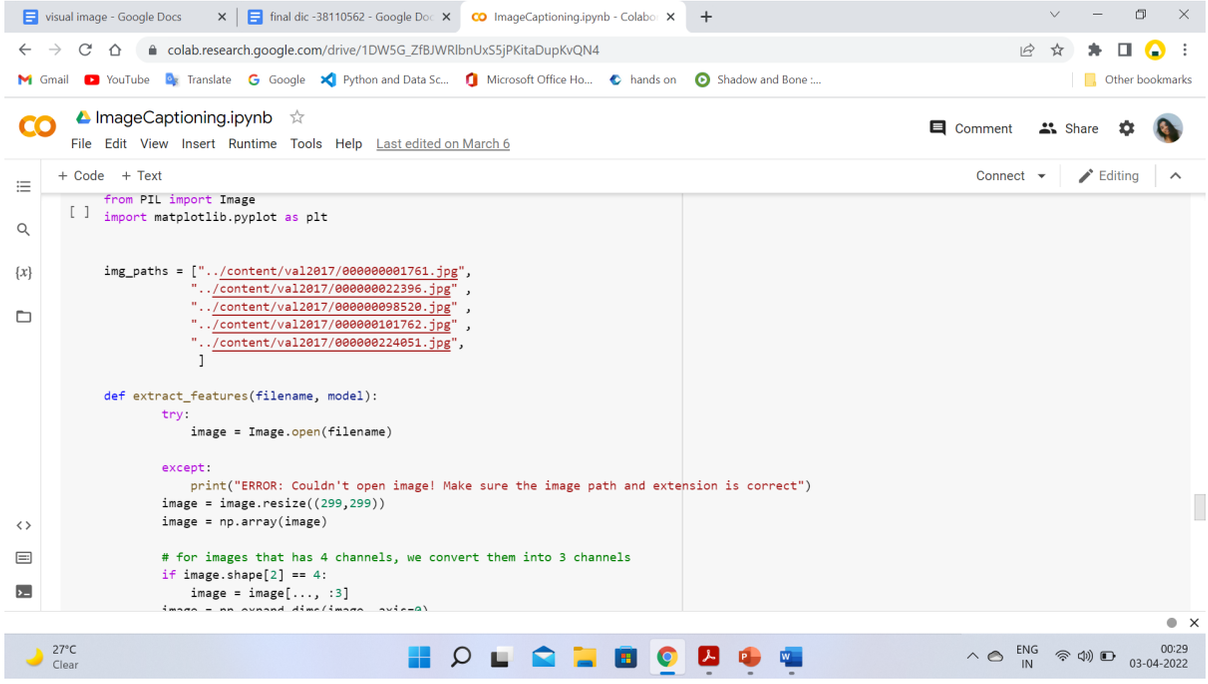
6.23 Making data generator



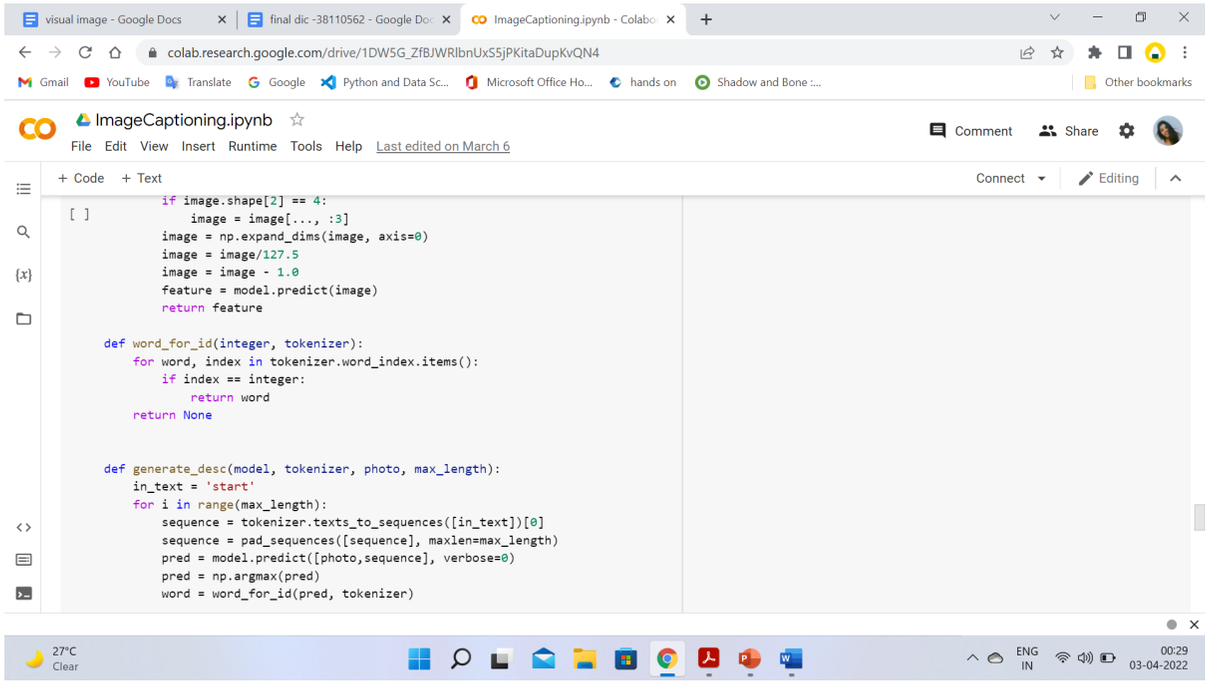
6.24 Making data generator



6.25 Train our model



6.26 Train our model (i.e., insert input path)



6.27 Train our model

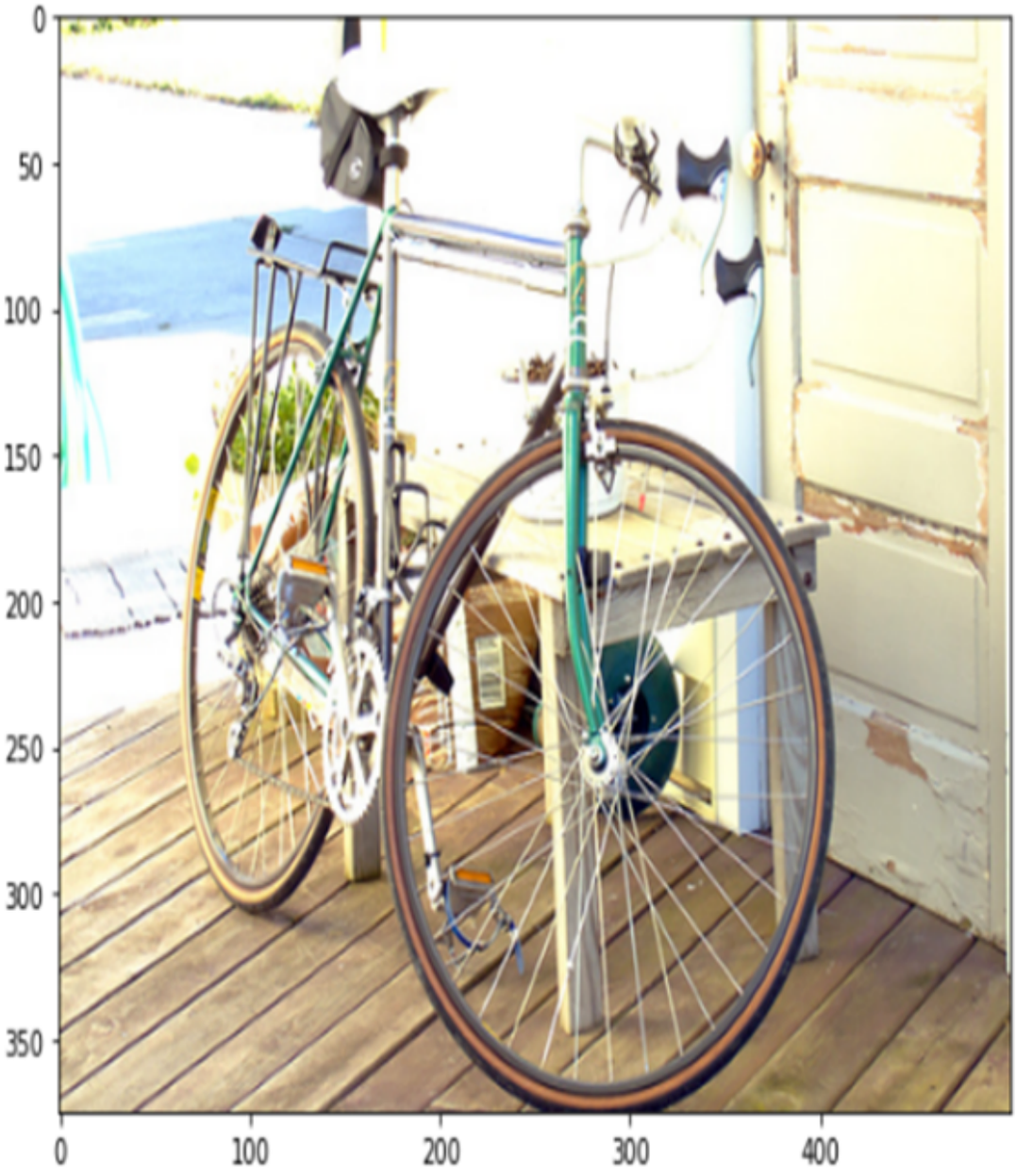
```
start a large airplane is parked on the runway end
<matplotlib.image.AxesImage at 0x7f453c993790>
```

6.28 Output for image input 1

A group of people watch water coming out of a fire hydran
t.

6.29 Output for image input 2

A bike rests on a deck near the beach.



6.30 Output for image input 3

```
start a man is riding a bike on a sidewalk end
```

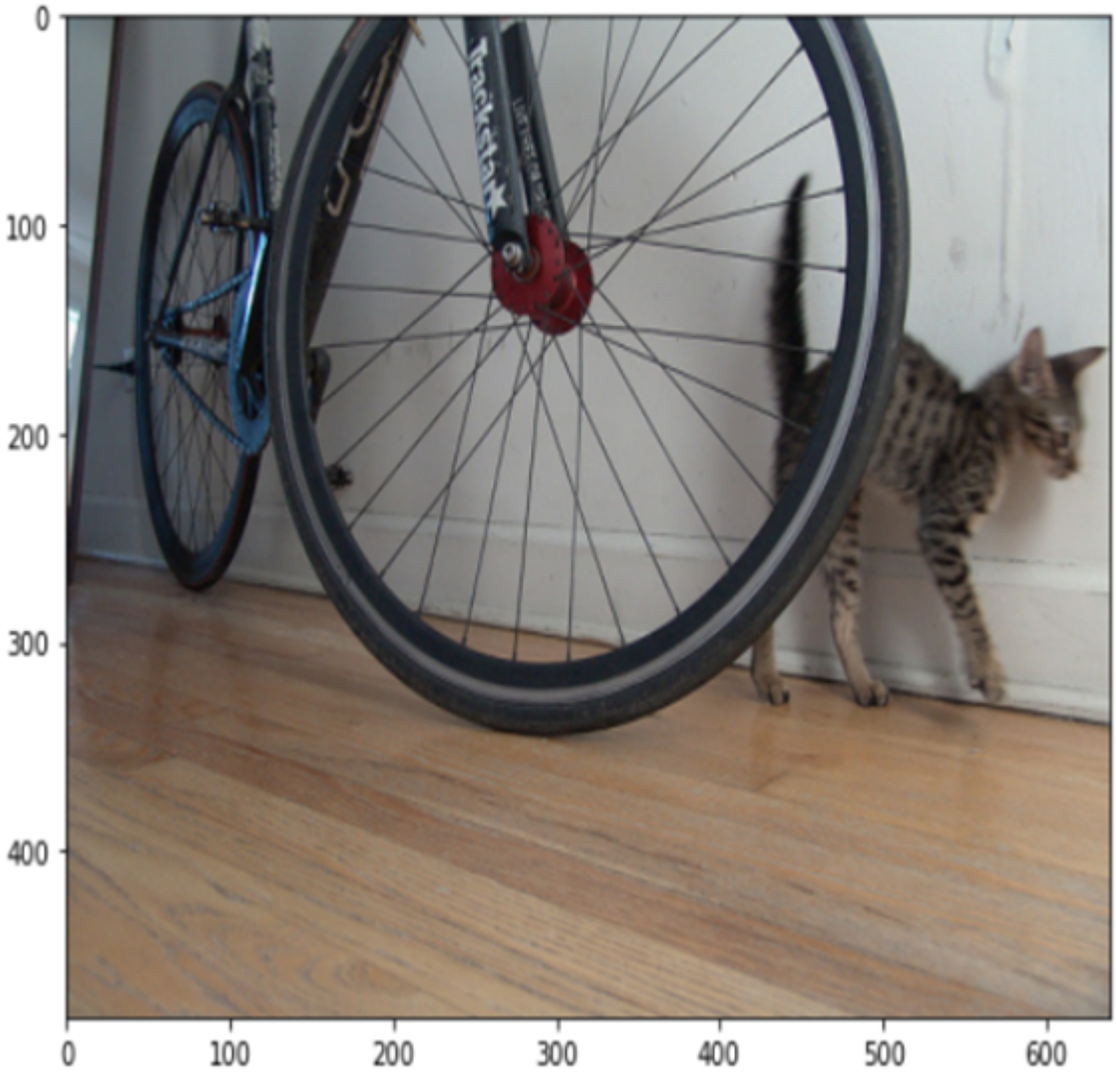
```
:
```

```
<matplotlib.image.AxesImage at 0x7f75368fed10>
```

6.31 Output for image input 4

```
start a bike parked next to a parking meter end
```

```
<matplotlib.image.AxesImage at 0x7f75881f2a50>
```



6.32 Output for image input 5

```
A green bus is on the street and it has a bicycle on the f
ront rack.
```

6.33 Output for image input 6

CHAPTER 7

SUMMARY AND CONCLUSION

7.1 Summary

We combined all components of the image caption generation problem in this overview, addressed the model framework proposed in recent years to handle the description task, concentrated on the algorithmic essence of various attention methods, and summarized how the attention mechanism is implemented. The huge datasets and evaluation criteria that are regularly utilized in practice are summarized. Despite the fact that image captioning can be used for image retrieval [92], video caption [93, 94], and video movement [95], and a wide range of image caption systems are currently available, experimental results suggest that this task still requires higher performance systems and improvement.

7.2 Conclusion

The CNN-LSTM model was created to automatically generate captions for the input images. This concept can be used in a wide range of situations. We learned about the CNN model, and LSTM models, and how to overcome previous limitations in the field of graphical image captioning by building a CNN-LSTM model capable of scanning and extracting information from any input image and transforming it into a single line sentence in natural language English.

The algorithm attention and how the attention mechanism is used were the main topics of discussion. I was able to successfully create a model that is a major improvement above the earlier image caption generator.

APPENDIX

SOURCE CODE:

```
!pip install CocoDataset==0.1.2
!wget
http://images.cocodataset.org/annotations/annotations_trainval2017.zip
!unzip /content/annotations_trainval2017.zip
!wget http://images.cocodataset.org/zips/train2017.zip
!unzip /content/train2017.zip
!wget http://images.cocodataset.org/zips/val2017.zip
!unzip /content/val2017.zip
!pip install pycocotools
from pycocotools.coco import COCO # COCO python library
import numpy as np
import skimage.io as io
import matplotlib.pyplot as plt
import pylab
import random
import string
import cv2
import os
from pickle import dump, load
import json
import nltk
nltk.download("stopwords")
from nltk.corpus import stopwords
import tensorflow as tf
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.layers import Embedding, LSTM, Dense,
Bidirectional, Input, Dropout, Attention
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.applications.xception import Xception,
preprocess_input
from tensorflow.keras.preprocessing.image import load_img, img_to_array
from tensorflow.keras.utils import to_categorical
from keras.layers.merge import add
from tensorflow.keras.models import Model, load_model

from tqdm.notebook import tqdm
```

```
pylab.rcParams['figure.figsize'] = (8.0, 10.0)
coco=COCO("../content/annotations/instances_train2017.json")
cats = coco.loadCats(coco.getCatIds())
maincategories = list(set([cat['supercategory'] for cat in cats]))

print("Number of main categories: ", len(maincategories))
print("List of main categories: ", maincategories)
subcategories = [cat['name'] for cat in cats]

print("Number of sub categories: ", len(subcategories))
print("List of sub categories: ", subcategories)
catIds = coco.getCatIds(catNms=subcategories)

subcategories_Ids = dict()
for i in range(0,len(subcategories)):
    subcategories_Ids[subcategories[i]] = catIds[i]

print("Sub categories with IDs :",subcategories_Ids)
subcategories_imageIds = dict()

for i in range(0,len(catIds)):
    imgIds = coco.getImgIds(catIds=catIds[i])
    img = []
    for j in imgIds:
        img.append(j)
    subcategories_imageIds[subcategories[i]] = img

print("Sub categories with Image IDs :",len(subcategories_imageIds))
length_dict     =     {key:     len(value)     for     key,     value     in
subcategories_imageIds.items()}
print("Total images in each sub categories: ", length_dict)
train_cats          =          subcategories_imageIds['bicycle']          +
subcategories_imageIds['airplane']
imgIdss = coco.getImgIds(imgIds = train_cats)
print("Total Images: ", len(imgIdss))
fig = plt.gcf()
fig.set_size_inches(16, 16)

next_pix = imgIdss
random.shuffle(next_pix)

for i, img_path in enumerate(next_pix[0:12]):

    sp = plt.subplot(4, 4, i + 1)
    sp.axis('Off')
```

```
    img = coco.loadImgs(img_path)[0]
    I = io.imread(img['coco_url'])
    plt.imshow(I)

plt.show()
fig = plt.gcf()
fig.set_size_inches(16, 16)

for i, img_path in enumerate(next_pix[0:12]):

    sp = plt.subplot(4, 4, i + 1)
    sp.axis('Off')

    img = coco.loadImgs(img_path)[0]
    I = io.imread(img['coco_url'])
    plt.imshow(I)
          annIds   =   coco.getAnnIds(imgIds=img['id'],   catIds=catIds,
iscrowd=None)
    anns = coco.loadAnns(annIds)
    # print(anns)
    coco.showAnns(anns)


plt.show()
annFile="../content/annotations/person_keypoints_train2017.json"
coco_kps=COCO(annFile)
fig = plt.gcf()
fig.set_size_inches(16, 16)

for i, img_path in enumerate(next_pix[0:12]):

    sp = plt.subplot(4, 4, i + 1)
    sp.axis('Off')

    img = coco.loadImgs(img_path)[0]
    I = io.imread(img['coco_url'])
    plt.imshow(I)
        annIds   =   coco_kps.getAnnIds(imgIds=img['id'],   catIds=catIds,
iscrowd=None)
    anns = coco_kps.loadAnns(annIds)
    coco_kps.showAnns(anns)


plt.show()
```

```
annFile = "../content/annotations/captions_train2017.json"
coco_caps=COCO(annFile)
img = coco.loadImgs(next_pix[0])[0]
I = io.imread(img['coco_url'])
plt.imshow(I)
annIds = coco_caps.getAnnIds(imgIds=img['id']);
anns = coco_caps.loadAnns(annIds)
coco_caps.showAnns(anns)
plt.show()
img = coco.loadImgs(next_pix[1])[0]
I = io.imread(img['coco_url'])
plt.imshow(I)
annIds = coco_caps.getAnnIds(imgIds=img['id']);
anns = coco_caps.loadAnns(annIds)
coco_caps.showAnns(anns)
plt.show()
img = coco.loadImgs(next_pix[10])[0]
I = io.imread(img['coco_url'])
plt.imshow(I)
annIds = coco_caps.getAnnIds(imgIds=img['id']);
anns = coco_caps.loadAnns(annIds)
coco_caps.showAnns(anns)
plt.show()
print("Total images for training: ", len(imgIdss))
dataset = dict()
imgcaptions = []

for imgid in imgIdss:
    img = coco.loadImgs(imgid)[0]
    annIds = coco_caps.getAnnIds(imgIds=img['id']);
    anns = coco_caps.loadAnns(annIds)
    imgcaptions = []
    for cap in anns:

        # Remove punctuation
                cap  =  cap['caption'].translate(str.maketrans('',  '',
string.punctuation))

        # Replace - to blank
        cap = cap.replace("-"," ")

         # Split string into word list and Convert each word into lower
case
        cap = cap.split()
        cap = [word.lower() for word in cap]
```

```
          # join word list into sentence and <start> and <end> tag to
each sentence which helps
        # LSTM encoder-decoder model while training.

        cap = '<start> ' + " ".join(cap) + ' <end>'
        imgcaptions.append(cap)

    dataset[img['coco_url']] = imgcaptions


print("Length of Dataset: ",len(dataset))
print(dataset['http://images.cocodataset.org/train2017/000000047084.jpg
'])
#dataset
from itertools import chain
flatten_list        =        list(chain.from_iterable(dataset.values()))
#[[1,3],[4,8]] = [1,3,4,8]

tokenizer = Tokenizer(oov_token='<oov>') # For those words which are
not found in word_index
tokenizer.fit_on_texts(flatten_list)
total_words = len(tokenizer.word_index) + 1

print("Vocabulary length: ", total_words)
print("Bicycle ID: ", tokenizer.word_index['bicycle'])
print("Airplane ID: ", tokenizer.word_index['airplane'])

print("Image features length: ", len(image_features))
image_features['http://images.cocodataset.org/train2017/000000047084.jp
g'].shape
def dict_to_list(descriptions):
    all_desc = []
    for key in descriptions.keys():
        [all_desc.append(d) for d in descriptions[key]]
    return all_desc

def max_length(descriptions):
    desc_list = dict_to_list(descriptions)
    return max(len(d.split()) for d in desc_list)

max_length = max_length(dataset)
max_length
#create input-output sequence pairs from the image description.
```

```
def data_generator(descriptions, features, tokenizer, max_length):
    while 1:
        for key, description_list in descriptions.items():

            feature = features[key][0]
                        input_image,  input_sequence,  output_word  =
create_sequences(tokenizer, max_length, description_list, feature)
            yield ([input_image, input_sequence], output_word)


def create_sequences(tokenizer, max_length, desc_list, feature):
    X1, X2, y = list(), list(), list()

    # walk through each description for the image
    for desc in desc_list:

        # encode the sequence
        seq = tokenizer.texts_to_sequences([desc])[0]

        # split one sequence into multiple X,y pairs
        for i in range(1, len(seq)):

            # split into input and output pair
            in_seq, out_seq = seq[:i], seq[i]

            # pad input sequence
            in_seq = pad_sequences([in_seq], maxlen=max_length)[0]

            # encode output sequence
                                out_seq   =   to_categorical([out_seq],
num_classes=total_words)[0]

            # store
            X1.append(feature) # image features
            X2.append(in_seq)  # Caption input
            y.append(out_seq)  # Caption output

    return np.array(X1), np.array(X2), np.array(y)
from tensorflow.keras.utils import plot_model

# define the captioning model
def define_model(total_words, max_length):

    # features from the CNN model squeezed from 2048 to 256 nodes
    inputs1 = Input(shape=(2048,))
```

```
    fe1 = Dropout(0.5)(inputs1)
    fe2 = Dense(256, activation='relu')(fe1)

    # LSTM sequence model
    inputs2 = Input(shape=(max_length,))
    se1 = Embedding(total_words, 256, mask_zero=True)(inputs2)
    se2 = Dropout(0.5)(se1)
    se3 = LSTM(256)(se2)

    # Merging both models
    decoder1 = add([fe2, se3])
    decoder2 = Dense(256, activation='relu')(decoder1)
    outputs = Dense(total_words, activation='softmax')(decoder2)

    # tie it together [image, seq] [word]
    model = Model(inputs=[inputs1, inputs2], outputs=outputs)
    model.compile(loss='categorical_crossentropy', optimizer='adam')

    # summarize model
    print(model.summary())
    plot_model(model, to_file='model.png', show_shapes=True)

    return model
# train our model
print('Dataset: ', len(dataset))
print('Descriptions: train=', len(dataset))
print('Photos: train=', len(image_features))
print('Vocabulary Size:', total_words)
print('Description Length: ', max_length)
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt


img_paths = ["../content/val2017/000000001761.jpg",
             "../content/val2017/000000022396.jpg" ,
             "../content/val2017/000000098520.jpg" ,
             "../content/val2017/000000101762.jpg" ,
             "../content/val2017/000000224051.jpg",
              ]

def extract_features(filename, model):
        try:
            image = Image.open(filename)
```

```
        except:
            print("ERROR: Couldn't open image! Make sure the image path
and extension is correct")
        image = image.resize((299,299))
        image = np.array(image)

            # for images that has 4 channels, we convert them into 3
channels
        if image.shape[2] == 4:
            image = image[..., :3]
        image = np.expand_dims(image, axis=0)
        image = image/127.5
        image = image - 1.0
        feature = model.predict(image)
        return feature

def word_for_id(integer, tokenizer):
    for word, index in tokenizer.word_index.items():
        if index == integer:
            return word
    return None


def generate_desc(model, tokenizer, photo, max_length):
    in_text = 'start'
    for i in range(max_length):
        sequence = tokenizer.texts_to_sequences([in_text])[0]
        sequence = pad_sequences([sequence], maxlen=max_length)
        pred = model.predict([photo,sequence], verbose=0)
        pred = np.argmax(pred)
        word = word_for_id(pred, tokenizer)

        if word is None:
            break
        in_text += ' ' + word

        if word == 'end':
            break
    return in_text


#max_length = 46

#model = load_model('./models/model_0.h5')
xception_model = Xception(include_top=False, pooling="avg")
```

```
photo = extract_features(img_paths[0], xception_model)
img = Image.open(img_paths[0])
description = generate_desc(model, tokenizer, photo, max_length)
print("\n\n")
print(description)
plt.imshow(img)
photo = extract_features(img_paths[1], xception_model)
img = Image.open(img_paths[1])
description = generate_desc(model, tokenizer, photo, max_length)
print("\n\n")
print(description)
plt.imshow(img)
photo = extract_features(img_paths[2], xception_model)
img = Image.open(img_paths[2])
description = generate_desc(model, tokenizer, photo, max_length)
print("\n\n")
print(description)
plt.imshow(img)
photo = extract_features(img_paths[3], xception_model)
img = Image.open(img_paths[3])
description = generate_desc(model, tokenizer, photo, max_length)
print("\n\n")
print(description)
plt.imshow(img)
photo = extract_features(img_paths[4], xception_model)
img = Image.open(img_paths[4])
description = generate_desc(model, tokenizer, photo, max_length)
print("\n\n")
print(description)
plt.imshow(img)
```

REFERENCES

[1] R. Subash (November 2019): Automatic Image Captioning Using Convolution Neural Networks and LSTM.

[2] Seung-Ho Han, Ho-Jin Choi (2020): Domain-Specific Image Caption Generator with Semantic Ontology.

[3] Pranay Mathur, Aman Gill, Aayush Yadav, Anurag Mishra and Nand Kumar Bansode (2017): Camera2Caption: A Real-Time Image Caption Generator

[4] Simao Herdade, Armin Kappeler, Kofi Boakye, Joao Soares (June 2019): Image Captioning: Transforming Objects into words.

[5] Manish Raypurkar, Abhishek Supe, Pratik Bhumkar, Pravin Borse, Dr. Shabnam Sayyad (March 2021): Deep learning-based Image Caption Generator

[6] Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan (2015): Show and Tell: A Neural Image Caption Generator

[7] Jianhui Chen, Wenqiang Dong, Minchen Li (2015): Image Caption Generator based on Deep Neural Networks

[8] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. (2017): Bottom-up and top-down attention for image captioning.

[9] Jyoti Aneja, Aditya Deshpande, and Alexander G Schwing (2018): Convolutional image captioning.

[10] Shuang Bai and Shan An (2018): A survey on automatic image caption generation.

[11] D.Bahdanau, K.Cho, and Y.Bengio(2015): Neural machine translation by jointly learning to align and translate.

[12] K.Xu, J.Ba, K.Cho, and R.Salakhutdinov (2018): Show attend and tell: Neural image caption generator with visual attention.

[13] M.Pedersoli, T.Lucas, C.Schmid, and J.Verbeek (2017): Areas of attention for image captioning.

[14] H.R.Tavakoli,R.Shetty, B.Ali, and J.Laaksonen(2017): Paying attention to descriptions generated by image captioning models.

[15] A.Matthews, L.Xie, and X.He(2018): Sem Style-learning to generate stylized image captions using unaligned text.

[16] C.Park, B.Kim, and G.kim(2018): Towards personalized image captioning via multimodal memory networks.

[17] X.Chen, Ma Lin, W.Jiang, J.Yao, and W.Liu(June 2018): Regularizing RNNs for caption generation by reconstructing the past with the present.

[18] T.Yao, Y.Pan, Y.Li, Z.Qiu, and T.Mei (June 2016): Boosting image captioning with attributes.

[19] Deep Learning in big data Analytics: A comparative study-Scientific Figure on ResarchGate (2021).

[20] Kaustav et al. (June 2016): A Facial Expression Recognition System to

predict Emotion.

A. PLAGIARISM REPORT

Curiginal

Document Information

Analyzed document	Reasrch paper-Sruthi(38110562).docx (D122332040)
Submitted	2021-12-14T07:09:00.0000000
Submitted by	
Submitter email	veena_k7@rediffmail.com
Similarity	8%
Analysis address	veena_k7.veltec@analysis.urkund.com

Sources included in the report

W	URL: https://www.irjet.net/archives/V8/i10/IRJET-V8I10165.pdf Fetched: 2021-11-12T09:25:09.3670000	1
W	URL: https://arxiv.org/pdf/1810.04020 Fetched: 2019-10-12T12:30:04.6800000	5
W	URL: https://openaccess.thecvf.com/content_cvpr_2018/papers/Aneja_Convolutional_Image_Captioning_CVPR_2018_paper.pdf Fetched: 2021-01-08T16:58:04.1300000	2
W	URL: https://openaccess.thecvf.com/content_cvpr_2018/papers/Mathews_SemStyle_Learning_to_CVPR_2018_paper.pdf Fetched: 2020-07-08T09:51:14.4430000	1
W	URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7199544/ Fetched: 2020-08-24T15:59:12.1400000	1

B. JOURNAL PAPER

Visual Image Caption Generator using Deep Learning

Sruthi Ravi

Department of Computer Science and Engineering, Sathyabama University, Chennai, Tamil Nadu, India Guide: Dr. Venna. K (M.E, Ph.D.)

Abstract - ***The combination of computer vision and natural language processing in Artificial intelligence has sparked a lot of interest in research in recent years, thanks to the advent of deep learning. The context of a photograph is automatically described in English. When a picture is captioned, the computer learns to interpret the visual information of the image using one or more phrases. The ability to analyze the state, properties, and relationship between these objects is required for the meaningful description generation process of high-level picture semantics. Using CNN -LSTM architectural models on the captioning of a graphical image, we hope to detect things and inform people via text messages in this research. To correctly identify the items, the input image is first reduced to grayscale and then processed by a Convolution Neural Network (CNN). The COCO Dataset 2017 was used. The proposed method for blind individuals is intended to be expanded to include persons with vision loss to speech messages to help them reach their full potential and to track their intellect. In this study paper, we meticulously follow a variety of important concepts of image captioning and its standard processes, as this work develops a generative CNN-LSTM model that outperforms human baselines.***

Key Words**: CNN, LSTM, Image Captioning, Computer Vision, Natural Language Processing, Deep Learning**

1. INTRODUCTION

Every day, we are bombarded with photos in our surroundings, on social media, and in the news. Only humans are capable of recognizing photos. We humans can recognize photographs without their assigned captions, but machines require images to be taught first. The encoder-decoder architecture of Image Caption Generator models uses input vectors to generate valid and acceptable captions. This paradigm connects the worlds of natural language processing and computer vision. It's a job of recognizing and evaluating the image's context before describing everything in a natural language like English. Our approach is based on two basic models: CNN (Convolutional Neural Network) and LSTM (Long Short-Term Memory). CNN is utilized as an encoder in the derived application to extract features from the snapshot or image, and LSTM is used as a decoder to organize the words and generate captions. Image captioning can help with a variety of things, such as assisting the visionless with text-to-speech through real-time input about the scenario over a camera feed, and increasing social medical leisure by restructuring captions for photos in social feeds as well as spoken messages. Assisting children in recognizing chemicals is a step toward learning the language. Captions for every photograph on the internet can result in faster and more accurate authentic photograph exploration and indexing. Image captioning is used in a variety of sectors, including biology, business, the internet, and in applications such as self-driving cars wherein it could describe the scene around the car, and CCTV cameras where the alarms could be raised if any malicious activity is observed. The main purpose of this research article is to gain a basic understanding of deep learning methodologies.

2. ARCHITECTURE AND WORKING

1. *Image Captioning* Techniques

***CNN*-Convolutional Neural Systems (CNNs) are essential neural systems that can produce information with a certain shape, such as a 2D lattice, and CNNs are useful when working with images. It analyses**

images from left to right, from corner to corner, to extract major highlights from the image, and then combines the elements to define the image. It can handle images that have been interpreted, rotated, scaled, and changed. The Convolutional neural system is a deep learning calculation that takes in a picture of information, assigns value to distinct components/protests in the picture, and distinguishes it from other pictures.

When compared to other order calculations, ConvNet's needed pre-handling is less. Even though channels are hand-designed in rudimentary strategies, ConvNets is capable of learning these channels/highlights with proper preparation. The curving system's shape is inspired by the way the visual cortex is organized and is similar to the neural network design found within the human brain. Singular neurons respond to upgrades in a small area of the visual field called the open field. The collection of such fields encompasses the totality of visual regions.

CNN: Architecture & Design When it comes to interpreting huge photos and videos, a pure primitive neural network, in which all neurons in one layer merge with all neurons in the next layer, is inefficient. The range of restriction using an acknowledged neural system for a regular size picture with multiple picture pieces called pixels and 3-tone colors (RGB i.e. red color, green color, blue color) will be in the thousands, resulting in overfitting.

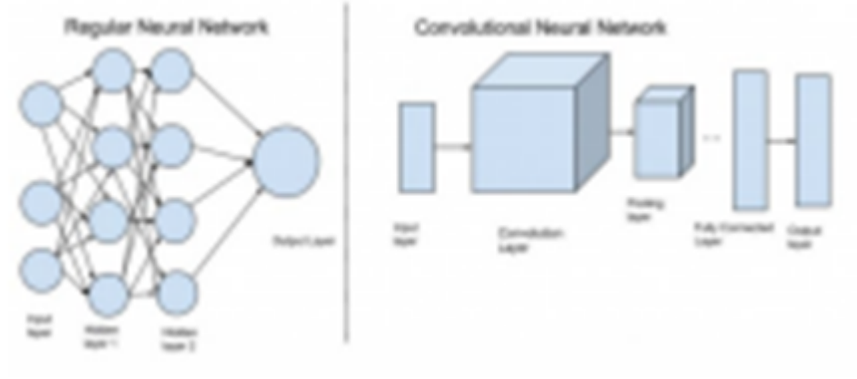


Fig.1 CNN Architecture

CNN uses a 3D arrangement in which each adjustment of neurons breaks down a little area or "highlight" of the picture to constrain effective quantities of constraints & recognition of the neural system on significant pieces of the picture. Rather than all neurons skipping to the next brain layer, each group of neurons spends a significant amount of time differentiating one aspect of the image, such as a nose, left ear, mouth, or leg. The final result is a point of scope, demonstrating how plausible each of the skills is chosen as a member of the class.

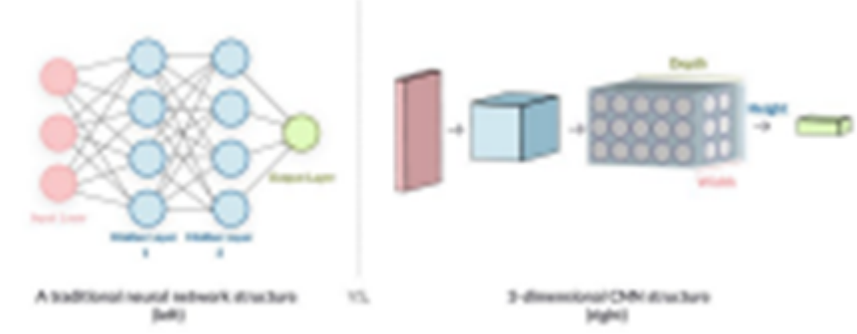
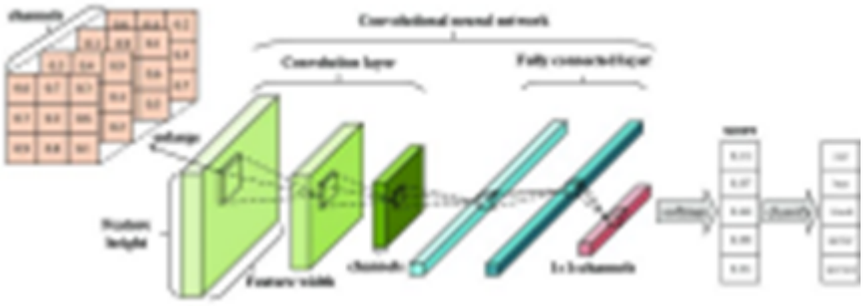


Fig.2 Working of CNN

How does CNN work ?

As previously discussed, a fully connected neural network, in which every input in the preceding layers is connected to every input in the following layers, is useful for the task at hand. Along these lines, CNN suggests

that the neurons in a cell may be connected to a specific cell area before it, rather than all the neurons in a cell in the same way.



This reduces the complexity of the neural network and the amount of computing power required. As seen on a new computer, using a standard image with numbers at each pixel. When we compare two photos in general, we look at the pixel values of each pixel. This technique only works when comparing two identical photos; when comparing different images, the comparison fails. Image comparison is done piece by piece on CNN.

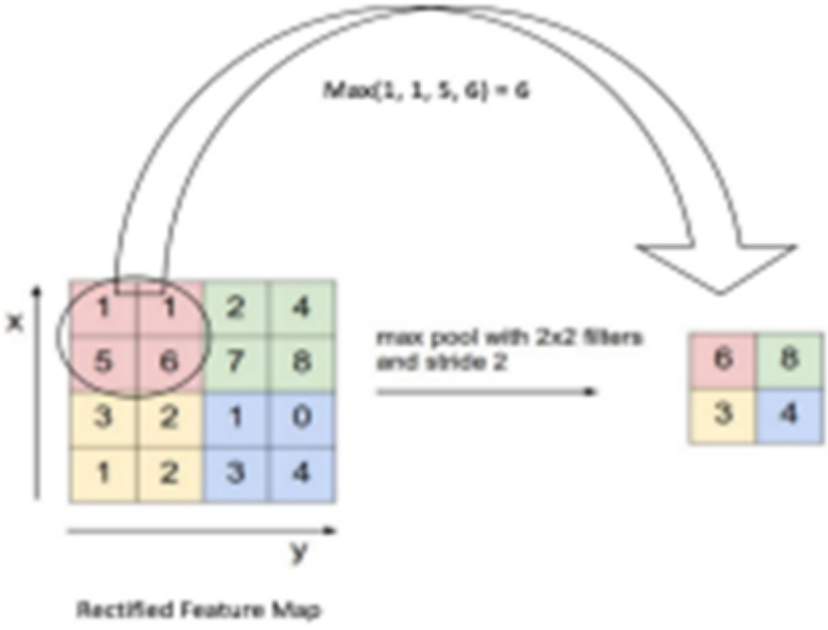


Fig.3 Feature map of CNN picture

The fundamental reason for utilizing the CNN method is that it is the only algorithm that accepts photographs as input and draws a feature map based on the input pictures, i.e. classifying each pixel based on similarity and differences. The CNN identifies the pixels and generates a feature map, which is a matrix. A feature map is a grouping of comparable pixels into a distinct category. These matrices are crucial in determining the essence of the object in the input image.

The first layer, Convolutional, and the second layer, Pooling, are practiced several times depending on the image to obtain dense information about the image. Fully Connected is the third layer. This layer is responsible for classification. It divides the pixels into groups based on their similarities and differences. Classification is carried out to an extreme degree to extract the essence of the image and aid in the identification of objects, people, and things, among other things.

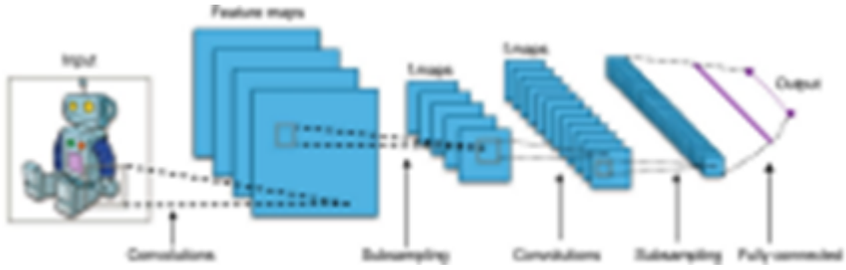


Fig.4 Layers of the scanned picture

These layers aid CNN in locating and identifying features in the image. The image of fixed length inputs is turned into fixed-size outputs by extracting important features.

LSTM-Long Short Term Memory.LSTM is a critical component in the Deep Learning discipline of recurrent neural networks. The unique feature of LSTM is that it can not only store the input data but also make predictions about forthcoming datasets using its data. This LSTM network saves the data for a specific period and then predicts or assigns future values to the data based on that. This is the primary reason why LSTM is preferred over regular RNN.

Problem with RNNs (Recurrent Neural Networks)?

RNNs are a type of deep learning rule set that is used to cope with a variety of difficult or sophisticated computer tasks such as item classification and speech recognition. RNNs are used to address a variety of activities that occur in sequence, with each situation's knowledge based entirely on statistics from previous scenarios.

To put it another way, we plan to favor RNNs with larger data sets and greater capabilities. This RNN may be used to solve a variety of real-world issues, such as inventory forecasting and speech recognition reinforcement. However, due to the Vanishing Gradient problem, RNNs are not used to solve real-world problems.

Why LSTM over RNNs?

We will use Long short-term memory (LSTM), which is a subset of RNNs, to tackle the problem of Vanishing Gradient. The main goal of LSTM is to solve the problem of Vanishing Gradients. The unique feature of LSTM is that it can keep data values for long periods, allowing it to address the vanishing gradient problem.

2. *Working*

We shall summarise the two distinct architectures to automatically generate construct an image caption generating model. It's also known as the CNN-LSTM model. So, to get the captions for the input photographs, we'll use these two architectures.

CNN was used to extract the most important features from the input image. To do so, we've used Xception, a pre-trained model for our consideration.

The LSTM has been utilized to store and analyze the data or features from the CNN model, as well as to assist in the production of a good caption for the image.

Python was used to bring this art to life.

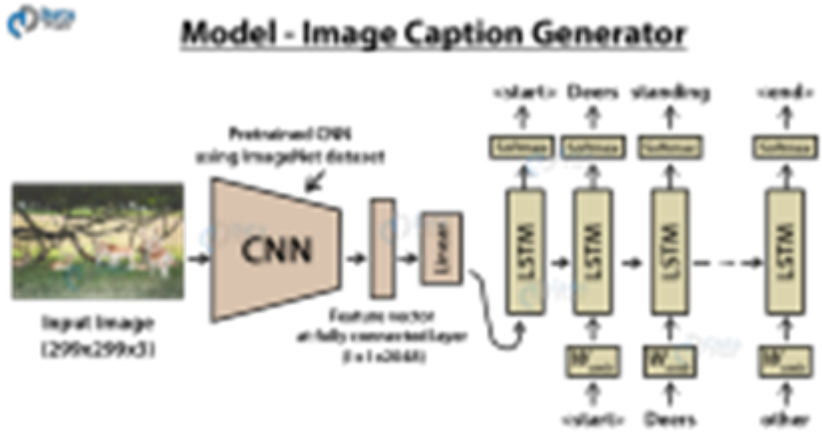


Fig.7 CNN-LSTM model

1. A gray-scale image is processed through CNN to identify the objects.
2. CNN scans images left-right, and top-bottom and extracts important image features. By applying various layers like Convolutional, Pooling, Fully Connected, and thus using the activation function, we successfully extracted features of every image.

3. It is then converted to LSTM.
4. Using the LSTM layer we try to predict what the next word could be and then proceeds to generate a sentence describing the image.

C. *Accuracy*

CNN is an extractor that extracts features from the given image.

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

The system accuracy will be measured using the standard equation.

3. RESULTS AND DISCUSSION

The Dataset that we used for our research is called "COCO Dataset 2017" and is available online. The data was pre-processed to make it suitable for future analysis and work. It consists of 12 main sorts of categories, each with 80 potential sub-categories.

Each subcategory has a collection of photographs as well as five captions for each one. The system performance was evaluated using the general confusion matrix. All of the models' results are listed here, along with their projections. Over the course of 30 iterations, a total of 130 iterations were completed.

4. CONCLUSION

The CNN-LSTM model was created to automatically generate captions for the input images. This concept can be used in a wide range of situations. We learned about the CNN model, and LSTM models, and how to overcome previous limitations in the field of graphical image captioning by building a CNN-LSTM model capable of scanning and extracting information from any input image and transforming it into a single line sentence in natural language English.

The algorithm attention and how the attention mechanism is used were the main topics of discussion. I was able to successfully create a model that is a major improvement above the earlier image caption generator.

5. FUTURE SCOPE

I'd like to train our model on a larger dataset with a greater number of photographs in the future. The captions generated should be in a range of languages. Larger datasets and alternative CNN architectures, such as LeNet, AlexNet, GoogLeNet, ResNet, and others, were used to train and evaluate the model.

Also, I'd like to use this model with a bigger audience, including blind individuals and a CCTV crew. Using IoT technology such as Arduino kits and cameras.

6. REFERENCES

[1] R. Subash (November 2019): Automatic Image Captioning Using Convolution Neural Networks and LSTM.

[2] Seung-Ho Han, Ho-Jin Choi (2020): Domain-Specific Image Caption Generator with Semantic Ontology.

[3] Pranay Mathur, Aman Gill, Aayush Yadav, Anurag Mishra and Nand Kumar Bansode (2017): Camera2Caption: A Real-Time Image Caption Generator

[4] Simao Herdade, Armin Kappeler, Kofi Boakye, Joao Soares (June 2019): Image Captioning: Transforming Objects into words.

[5] Manish Raypurkar, Abhishek Supe, Pratik Bhumkar, Pravin Borse, Dr. Shabnam Sayyad (March 2021): Deep learning-based Image Caption Generator

[6] Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan (2015): Show and Tell: A Neural Image Caption Generator

[7] Jianhui Chen, Wenqiang Dong, Minchen Li (2015): Image Caption Generator based on Deep Neural Networks

[8] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang.(2017): Bottom-up and top-down attention for image captioning.

[9] Jyoti Aneja, Aditya Deshpande, and Alexander G Schwing(2018): Convolutional image captioning.

[10] Shuang Bai and Shan An (2018): A survey on automatic image caption generation.

[11] D.Bahdanau, K.Cho, and Y.Bengio(2015): Neural machine translation by jointly learning to align and translate.

[12] K.Xu, J.Ba, K.Cho, and R.Salakhutdinov (2018): Show attend and tell: Neural image caption generator with visual attention.

[13] M.Pedersoli, T.Lucas, C.Schmid, and J.Verbeek (2017): Areas of attention for image captioning.

[14] H.R.Tavakoli,R.Shetty, B.Ali, and J.Laaksonen(2017): Paying attention to descriptions generated by image captioning models.

[15] A.Matthews, L.Xie, and X.He(2018): Sem Style-learning to generate stylized image captions using unaligned text.

[16] C.Park, B.Kim, and G.kim(2018): Towards personalized image captioning via multimodal memory networks.

[17] X.Chen, Ma Lin, W.Jiang, J.Yao, and W.Liu(June 2018): Regularizing RNNs for caption generation by reconstructing the past with the present.

[18] T.Yao, Y.Pan, Y.Li, Z.Qiu, and T.Mei (June 2016): Boosting image captioning with attributes.

[19] Deep Learning in big data Analytics: A comparative study-Scientific Figure on ResarchGate (2021).

[20] Kaustav et al. (June 2016): A Facial Expression Recognition System to predict Emotions.