A

Major Project

On

AUTOMATIC GESTURE RECOGNITION AND IMAGE CAPTION GENERATOR

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

B. KAWSHIK 197R1A05K0)

G. SAI MAHESH (197R1A05K9)

M.SAI KUMAR (197R1A05M8)

Under the Guidance of

J. NARASIMHA RAO

(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road,

Hyderabad-501401.

2019-2023

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATION

This is to certify that the Project entitled "**AUTOMATIC GESTURE RECOGNITION AND IMAGE CAPTION GENERATOR**" being submitted by **B.KAWSHIK (197R1A05K0),G.SAI MAHESH (197R1A05K9) &M.SAI KUMAR (197R1A05M8**) in partial fulfillment of the requirements for the award of degree B. TECH Computer science and engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bondafide work carried out by them under our guidance and supervision during the year 2022-23.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

J. NARASIMHA RAO
(Associate Professor)
INTERNAL GUIDE

DR.A. RAJI REDDY
DIRECTOR

Dr. K. Srujan Raju
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on ____________________

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **J. Narasimha Rao,** Associate Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **Dr. Punyaban Patel, Ms. Shilpa, Dr. T. Subha Mastan Rao & J. Narasimharao** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion ofthis project.

B. KAWSHIK 197R1A05K0)
G.SAI MAHESH (197R1A05K9)
M.SAI KUMAR (197R1A05M8)

ABSTRACT

Caption generation is a difficult synthetic intelligence problem wherein a textual description maybe generated for a given photograph/video. On every occasion when a photograph/video seems in front of people, their mind is capable of annotating or labeling it, however how can computer structures procedure and label it with a noticeably relevant and correct caption? So we will train the computers using deep learning algorithms and make it easy. With advanced deep learning techniques, accessibility of big datasets and computer power we can build an efficient model to generate captions. The strategies used right here are **LSTM (Long ShortTerm Memory) and CNN (Convolutional Neural Networks).** Feature extraction is done first and then captions are generated. The flickr_8k dataset is used for training the model. The dataset which we are using contains 8000 images and each image is mapped with five different captions

DOMAIN: DEEP LEARNING

LIST OF FIGURES/TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1.INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

Using Natural Language Processing (NLP) to make a computer system discover and describe items is an age-old artificial intelligence problem. Image caption generation is an image processing activity that recognises the context of an image and describes it in plain language. This image caption generator will generate the caption from a trained model that has been trained using algorithms and on a big dataset based on the picture file we provide or upload. It may be a difficult challenge to automatically describe the substance of a photograph using properly constructed English sentences, but it might have a significant impact, such as assisting visually impaired persons in better understanding the image online. We come across a wide variety of things every day.

1.2 PROJECT PURPOSE

Captions for every picture at the net can result in quicker and descriptively correct photos searches and indexing. Since researchers began working on image recognition, it has been clear that simply providing the names of the items recognised does not make the same kind of impression as a detailed human-like description. Natural language descriptions will remain a problem to be solved as long as machines do not think, speak, or behave like humans. Image captioning is used in a variety of sectors, including biomedicine, trade, internet searching, and the military. Instagram, Facebook, and other social media platforms can produce captions for photos automatically.

1.3 PROJECT FEATURES

Image caption generator is a project that uses computer vision and natural language processing ideas to comprehend the context of an image and explain it in natural language like English. We might have utilised CNN (Convolutional Neural Networks) and LSTM to generate captions for this Python-based application (Long Short Term Memory). The photograph features will be taken from VGG16, a CNN version trained on the image net dataset, and then fed into the LSTM model, which will be responsible for creating the photograph captions. Convolutional neural networks are deep neural networks that have been customised to process data in the form of a second matrix Photographs are easy. It is able to handle the photos that have been translated, circled, scaled and modifications in angle. Long ShortTerm Memory (LSTM) is a form of RNN (Recurrent Neural Network) that excels at sequence prediction. Based on the prior textual content, we can estimate what the next phrase will be. LSTM may perform relevant statistics throughout the input processing and, using an overlook gate, it can eliminate irrelevant data. So that it will be easy for the user to recognise the image in less time. The flickr_8k dataset is used for training the model. The dataset which we are using contains 8000 images and each image is mapped with five different captions.These applications in image captioning have important theoretical and practical research value. Image captioning is a more complicated but meaningful task in the age of artificial intelligence. Given a new image, an image captioning algorithm should output a description about this image at a semantic level. In this an Image caption generator, basis on our provided or uploaded image file It will generate the caption from a trained model which is trained using algorithms and on a large dataset. The main idea behind this is that users will get automated captions when we use or implement it on social media or on any applications.

2.SYSTEM ANALYSIS

2.SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analysed . The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

A general statement of the proposed model is to generate the relevant natural language caption to the given input image, instead of just describing a single target object the model detects multiple target objects for generating grammatically correct caption. We have used flickr_8k data set, the working of the model.

2.2 EXISTING SYSTEM

Image captioning has gotten a lot of attention recently, especially in the natural language area. The necessity for natural language image descriptions with context is critical. While this may appear far-fetched, recent advancements in domains like neural networks, computer vision, and natural language processing have paved the path for effectively characterising images, or capturing their visually grounded meaning.We're using cutting-edge approaches like Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), as well as associated image and humanperceived description datasets, to do this. We show that our alignment approach works in retrieval tests on datasets like Flickr.

2.2.1DRABACKS OF EXISTING SYSTEM

Following are the disadvantages of existing system:

- Can describe only one single Target object. So that, we can't get accurate correct caption.
- Not used advance approaches at present.
- It only gives caption to single frame.
- It is only for images.

2.3 PROPOSED SYSTEM

The proposed model is to generate the relevant natural language caption to the given input image, instead of just describing a single target object the model detects multiple target objects for generating grammatically correct caption .We have used flickr_8k data set .

The working of the model involves the following steps:

1.The given input image is pre-processed.

2. **Convolution neural network** (CNN) is used to extract features from the provided input image.

3.The information from the CNN is used by the LSTM for generating the relevant caption for the given image.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

- In this project, instead of just describing a single target object, this model detects multiple target objects and generating grammatically correct caption.
- Higher accuracy &stability.
- Quicker time to unlock a device.
- Can convey a range of Accurate emotions.

2.4 FEASIBILITY STUDY

The feasibility of the project is analysed in this phase and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on a project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require. The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication that the system is economically possible for development

2.4.2 TECHNICAL FEASIBILITY

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

2.4.3 BEHAVIORAL FEASIBILITY

This includes the following questions:

● Is there sufficient support for the users?

● Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioural aspects are considered carefully and conclude that the project is behaviourally feasible

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system.

The following are some hardware requirements:

- Processor – i5 and above (64-bit OS).
- Memory – 4GB RAM (Higher specs are recommended for high performance).
- Input devices – Keyboard, Mouse

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system.

The following are some software requirements:

- Windows/Mac.
- Kaggle Notebook
- Python3
- CNN and LSTM algorithm
- NumPy, Keras, Tensor Flow libraries

3.ARCHITECTURE

3.ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final output.

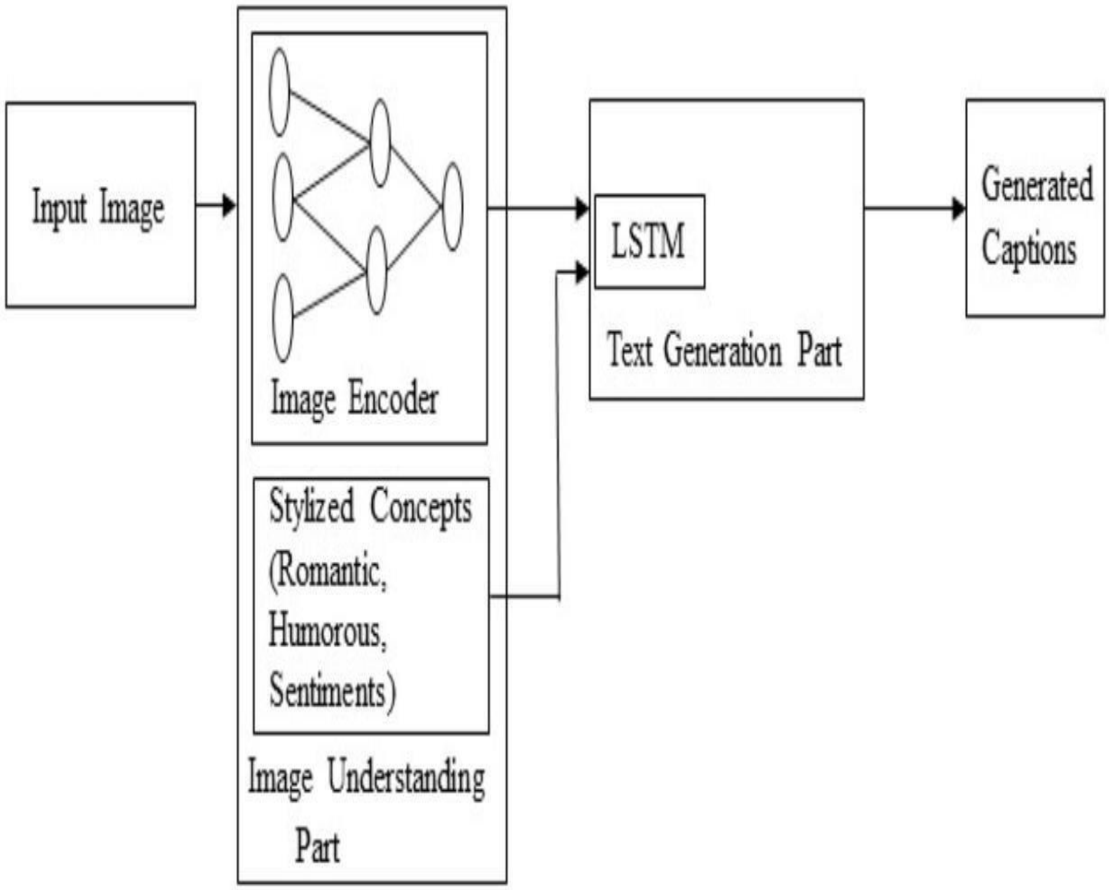


Figure 3.1: Project Architecture Automatic Gesture Recognition And Image Caption Generator

3.2 DESCRIPTION

This project is totally based upon to generate the relevant natural language caption to the given input image, instead of just describing a single target object the model detects multiple target objects for generating grammatically correct caption. We have used flickr_8k data set .The given input image is pre-processed. **Convolution neural network** (CNN) is used to extract features from the provided input image .The information from the **CNN** is used by the **LSTM** for generating the relevant caption for the given image. The proposed model is to generate the relevant natural language caption to the given input image, instead of just describing a single target object the model detects multiple target objects for generating grammatically correct caption. We have used flickr_8k data set, the working of the model. These applications in image captioning have important theoretical and practical research value. Image captioning is a more complicated but meaningful task in the age of artificial intelligence. Given a new image, an image captioning algorithm should output a description about this image at a semantic level. In this an Image caption generator, basis on our provided or uploaded image file It will generate the caption from a trained model which is trained using algorithms and on a large dataset. The main idea behind this is that users will get automated captions when we use or implement it on social media or on any applications.

3.3 USE CASE DIAGRAM

In the use case diagram, we have basically one actor who is the user in the trained model. A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

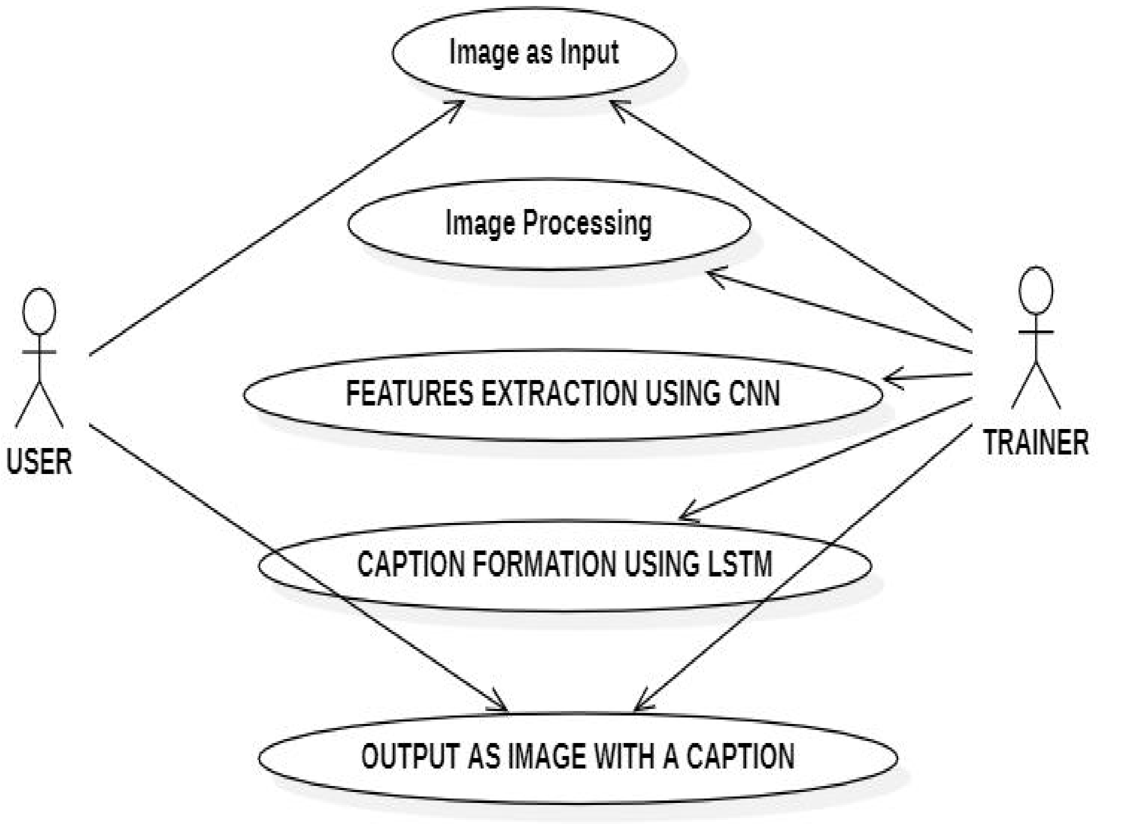


Figure 3.2: Use Case Diagram Automatic Gesture Recognition And Image Caption Generator

3.4 CLASS DIAGRAM

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations(methods),and the relationships among objects.

Figure 3.3: Class Diagram Automatic Gesture Recognition And Image Caption Generator

3.5 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

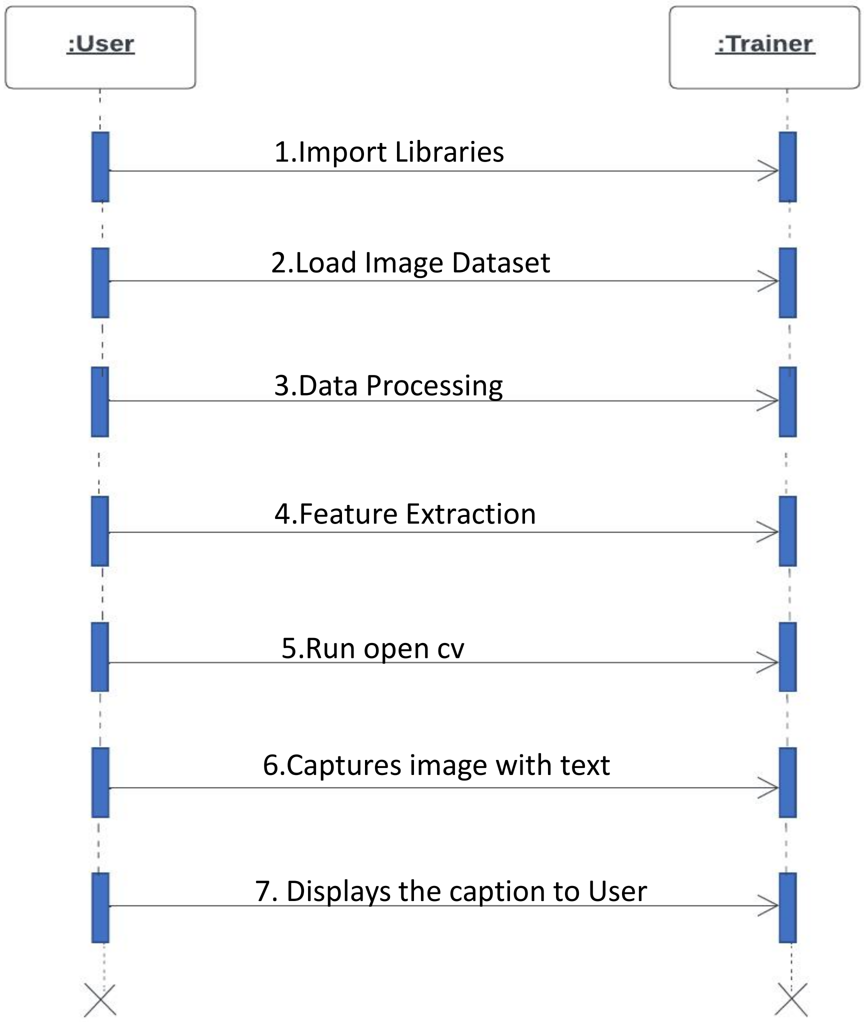


Figure 3.4: Sequence Diagram Automatic Gesture Recognition And Image Caption Generator

3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of work flows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

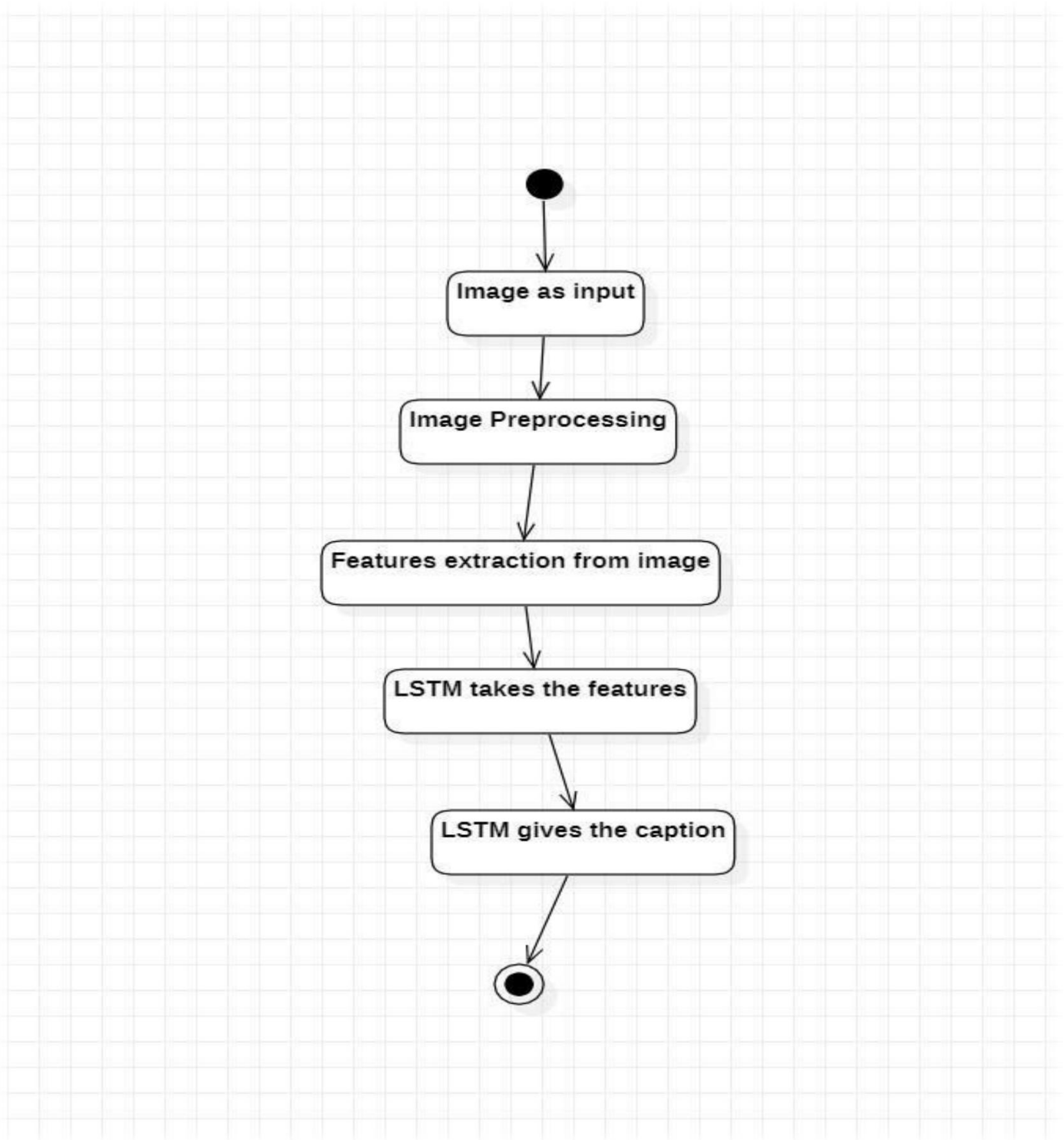


Figure 3.5: Activity Diagram Automatic Gesture Recognition And Image Caption Generator

4.IMPLEMENTATION

4.1 SAMPLE CODE

```
1)import os
import pickle
import numpy as np
from tqdm.notebook import tqdm

from tensorflow.keras.applications.vgg16 import VGG16, preprocess_input
from tensorflow.keras.preprocessing.image import load_img, img_to_array
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Model
from tensorflow.keras.utils import to_categorical, plot_model
from tensorflow.keras.layers import Input, Dense, LSTM, Embedding, Dropout, add


2)BASE_DIR = '/kaggle/input/flickr8k'
WORKING_DIR = '/kaggle/working'


Extract Image Features

3)# load vgg16 model
model = VGG16()
# restructure the model
model = Model(inputs=model.inputs, outputs=model.layers[-2].output)
# summarize
print(model.summary())

4)features = {}
directory = os.path.join(BASE_DIR, 'Images')

for img_name in tqdm(os.listdir(directory)):
  # load the image from file
  img_path = directory + '/' + img_name
  image = load_img(img_path, target_size=(224, 224))
  # convert image pixels to numpy array
  image = img_to_array(image)
  # reshape data for model
  image = image.reshape((1, image.shape[0], image.shape[1], image.shape[2]))
  # preprocess image for vgg
  image = preprocess_input(image)
  # extract features
  feature = model.predict(image, verbose=0)
  # get image ID
  image_id = img_name.split('.')[0]
```

```
    # store feature
    features[image_id] = feature

5)pickle.dump(features, open(os.path.join(WORKING_DIR, 'features.pkl'), 'wb'))

6)with open(os.path.join(WORKING_DIR, 'features.pkl'), 'rb') as f:
    features = pickle.load(f)
```

Load the Captions Data

```
7)with open(os.path.join(BASE_DIR, 'captions.txt'), 'r') as f:
    next(f)
    captions_doc = f.read()
8)mapping = {}
# process lines
for line in tqdm(captions_doc.split('\n')):
    # split the line by comma(,)
    tokens = line.split(',')
    if len(line) < 2:
        continue
    image_id, caption = tokens[0], tokens[1:]
    # remove extension from image ID
    image_id = image_id.split('.')[0]
    # convert caption list to string
    caption = " ".join(caption)
    # create list if needed
    if image_id not in mapping:
        mapping[image_id] = []
    # store the caption
    mapping[image_id].append(caption)

9)len(mapping)

10)def clean(mapping):
    for key, captions in mapping.items():
        for i in range(len(captions)):
            # take one caption at a time
            caption = captions[i]
            # preprocessing steps
            # convert to lowercase
            caption = caption.lower()
            # delete digits, special chars, etc.,
            caption = caption.replace('[^A-Za-z]', '')
            # delete additional spaces
            caption = caption.replace('\s+', ' ')
            # add start and end tags to the caption
```

```
            caption = 'startseq ' + " ".join([word for word in caption.split() if len(word)>1]) + '
endseq'
            captions[i] = caption

11)mapping['1000268201_693b08cb0e']


12)clean(mapping)

13)mapping['1000268201_693b08cb0e']


14)all_captions = []
for key in mapping:
    for caption in mapping[key]:
        all_captions.append(caption)

15)len(all_captions)

16)all_captions[:10]

17)tokenizer = Tokenizer()
tokenizer.fit_on_texts(all_captions)
vocab_size = len(tokenizer.word_index) + 1

18)vocab_size

19)max_length = max(len(caption.split()) for caption in all_captions)
max_length

20)image_ids = list(mapping.keys())
split = int(len(image_ids) * 0.90)
train = image_ids[:split]
test = image_ids[split:]

21)def data_generator(data_keys, mapping, features, tokenizer, max_length,
vocab_size, batch_size):
    # loop over images
    X1, X2, y = list(), list(), list()
    n = 0
    while 1:
        for key in data_keys:
            n += 1
            captions = mapping[key]
            # process each caption
            for caption in captions:
                # encode the sequence
```

```
            seq = tokenizer.texts_to_sequences([caption])[0]
            # split the sequence into X, y pairs
            for i in range(1, len(seq)):
                # split into input and output pairs
                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=vocab_size)[0]

                # store the sequences
                X1.append(features[key][0])
                X2.append(in_seq)
                y.append(out_seq)
        if n == batch_size:
            X1, X2, y = np.array(X1), np.array(X2), np.array(y)
            yield [X1, X2], y
            X1, X2, y = list(), list(), list()
            n = 0
```

Model Creation

```
22)inputs1 = Input(shape=(4096,))
fe1 = Dropout(0.4)(inputs1)
fe2 = Dense(256, activation='relu')(fe1)
# sequence feature layers
inputs2 = Input(shape=(max_length,))
se1 = Embedding(vocab_size, 256, mask_zero=True)(inputs2)
se2 = Dropout(0.4)(se1)
se3 = LSTM(256)(se2)

# decoder model
decoder1 = add([fe2, se3])
decoder2 = Dense(256, activation='relu')(decoder1)
outputs = Dense(vocab_size, activation='softmax')(decoder2)

model = Model(inputs=[inputs1, inputs2], outputs=outputs)
model.compile(loss='categorical_crossentropy', optimizer='adam')

plot_model(model, show_shapes=True)

23)epochs = 20
batch_size = 32
steps = len(train) // batch_size

for i in range(epochs):
```

```
    # create data generator
    generator = data_generator(train, mapping, features, tokenizer, max_length,
vocab_size, batch_size)
    # fit for one epoch
    model.fit(generator, epochs=1, steps_per_epoch=steps, verbose=1)
```

```
24)model.save(WORKING_DIR+'/best_model.h5')
```

Generate Captions for the Image

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

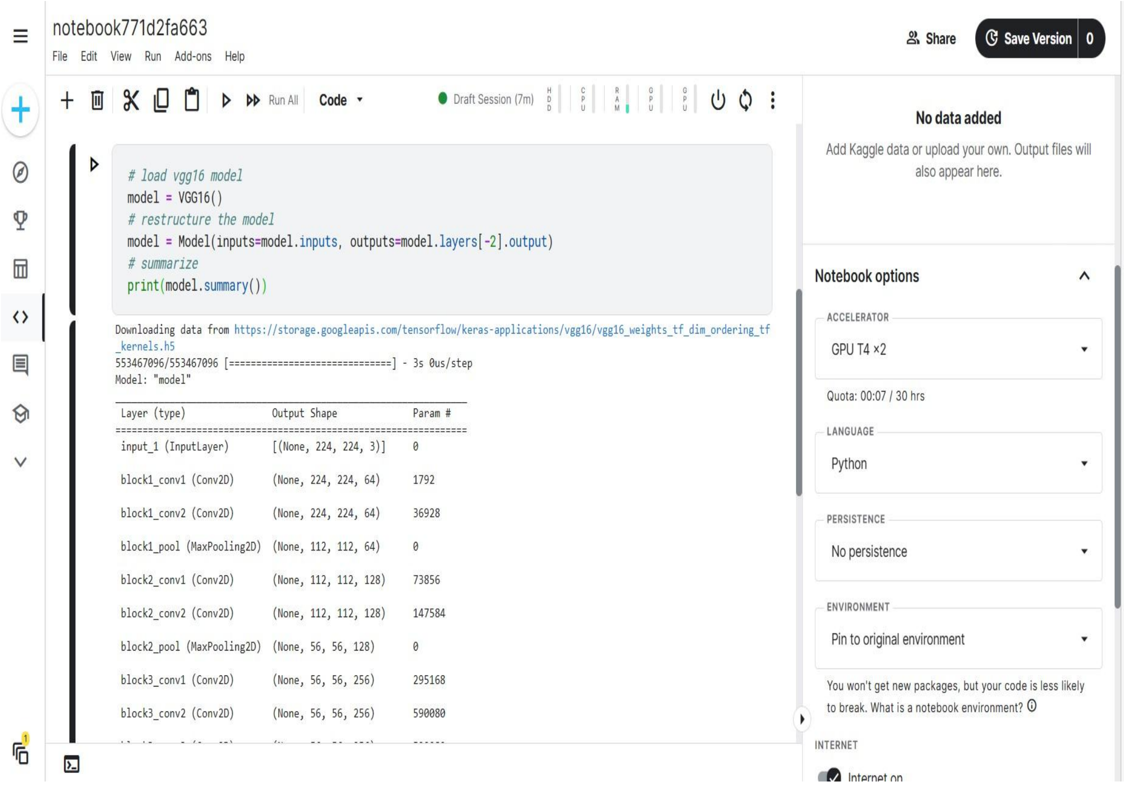
```
26)def predict_caption(model, image, tokenizer, max_length):
    # add start tag for generation process
    in_text = 'startseq'
    # iterate over the max length of sequence
    for i in range(max_length):
        # encode input sequence
        sequence = tokenizer.texts_to_sequences([in_text])[0]
        # pad the sequence
        sequence = pad_sequences([sequence], max_length)
        # predict next word
        yhat = model.predict([image, sequence], verbose=0)
        # get index with high probability
        yhat = np.argmax(yhat)
        # convert index to word
        word = idx_to_word(yhat, tokenizer)
        # stop if word not found
        if word is None:
            break
        # append word as input for generating next word
        in_text += " " + word
        # stop if we reach end tag
        if word == 'endseq':
            break

    return in_text
```

```
27)from nltk.translate.bleu_score import corpus_bleu
# validate with test data
actual, predicted = list(), list()
```

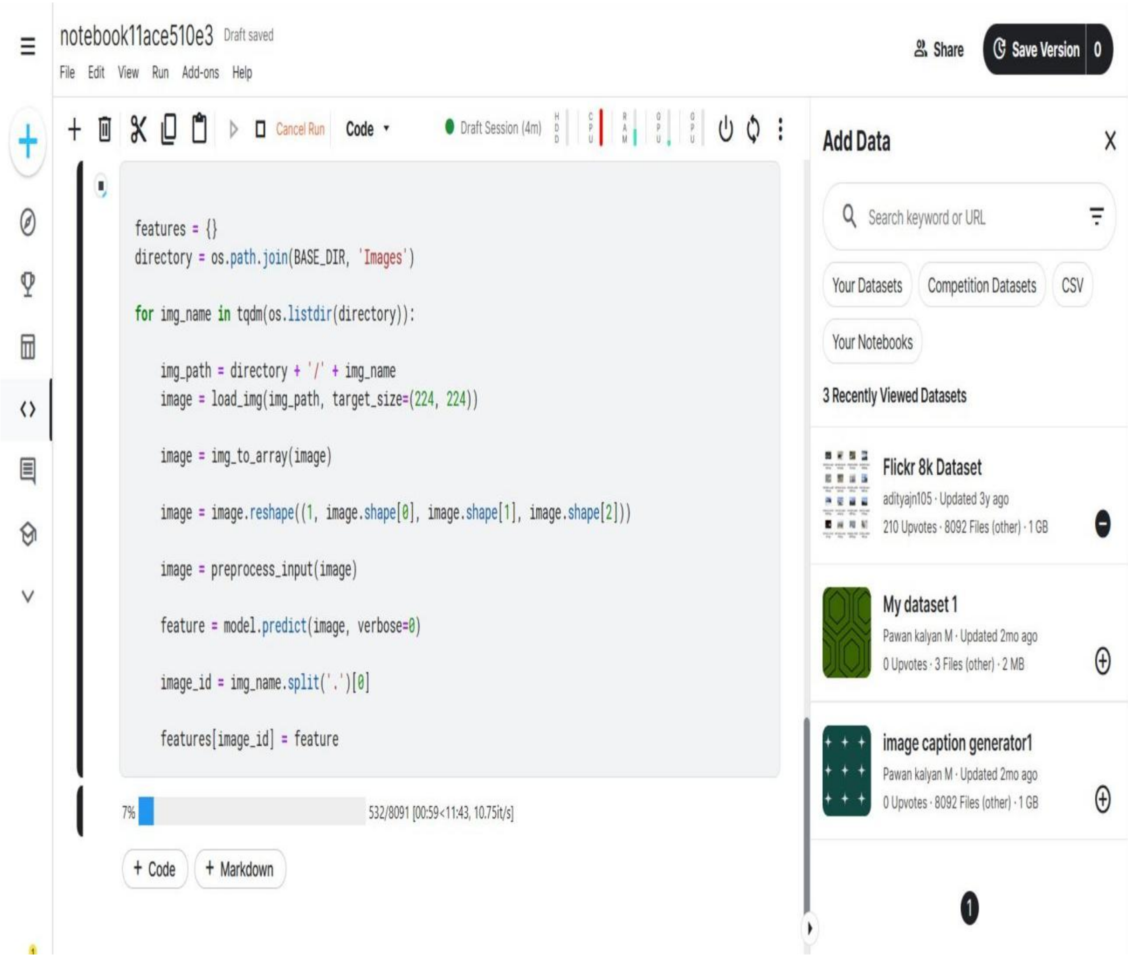
```
for key in tqdm(test):
    # get actual caption
    captions = mapping[key]
    # predict the caption for image
    y_pred = predict_caption(model, features[key], tokenizer, max_length)
    # split into words
    actual_captions = [caption.split() for caption in captions]
    y_pred = y_pred.split()
    # append to the list
    actual.append(actual_captions)
    predicted.append(y_pred)

# calcuate BLEU score
print("BLEU-1: %f" % corpus_bleu(actual, predicted, weights=(1.0, 0, 0, 0)))
print("BLEU-2: %f" % corpus_bleu(actual, predicted, weights=(0.5, 0.5, 0, 0)))

28)from PIL import Image
import matplotlib.pyplot as plt
def generate_caption(image_name):
    # load the image
    # image_name = "1001773457_577c3a7d70.jpg"
    image_id = image_name.split('.')[0]
    img_path = os.path.join(BASE_DIR, "Images", image_name)
    image = Image.open(img_path)
    captions = mapping[image_id]
    print('---------------------Actual---------------------')
    for caption in captions:
        print(caption)
    # predict the caption
    y_pred = predict_caption(model, features[image_id], tokenizer, max_length)
    print('--------------------Predicted--------------------')
    print(y_pred)
    plt.imshow(image)

29)generate_caption("1001773457_577c3a7d70.jpg")

30)generate_caption("1002674143_1b742ab4b8.jpg")
```

5. RESULTS

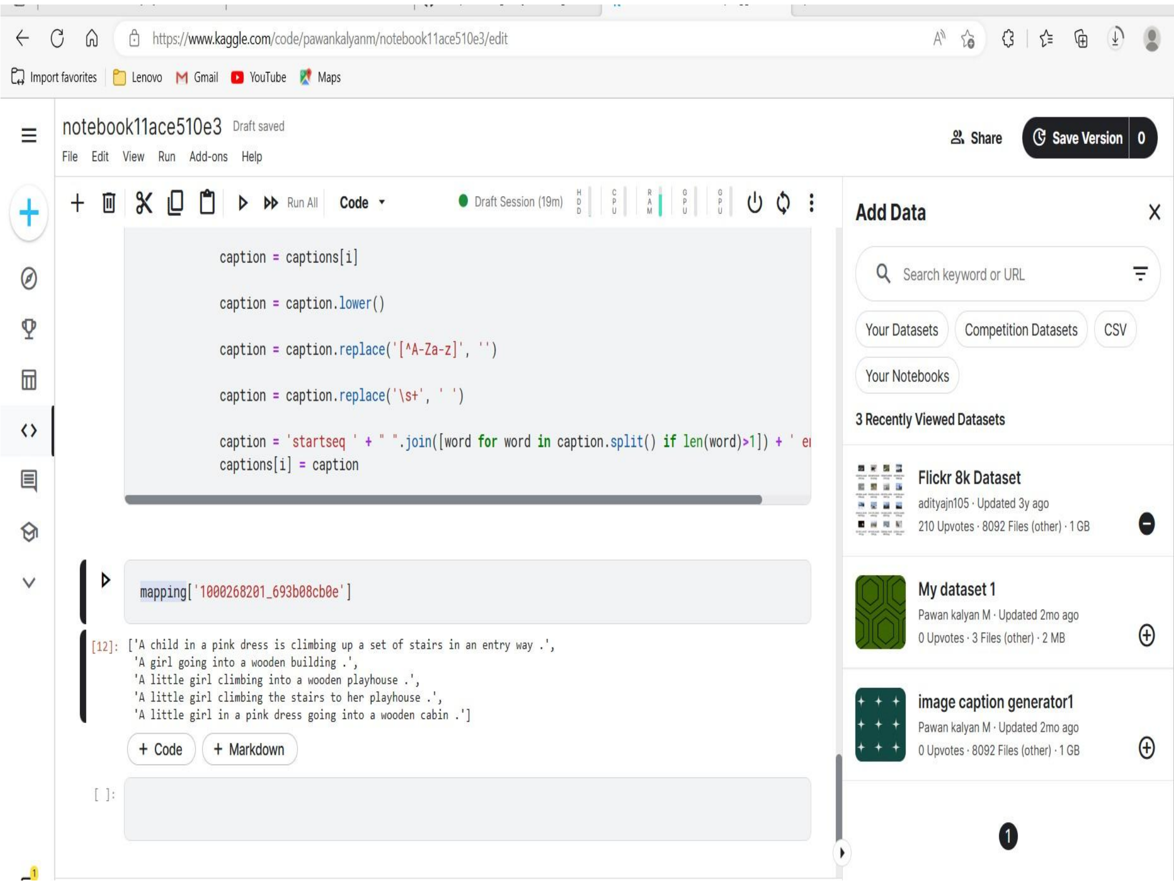
5. RESULTS



Screenshot 5.1: Exacting Features

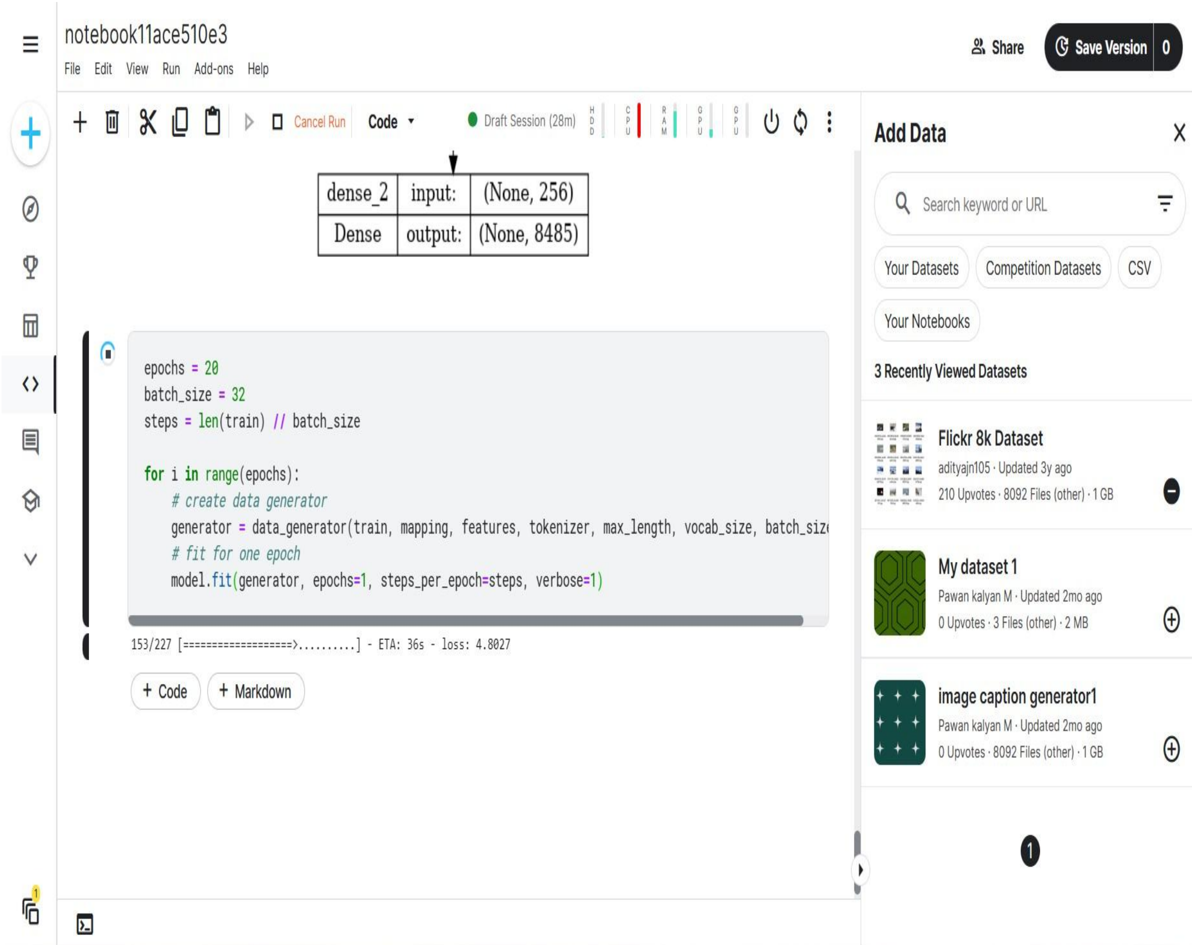


Screenshot 5.2: Exacting Images

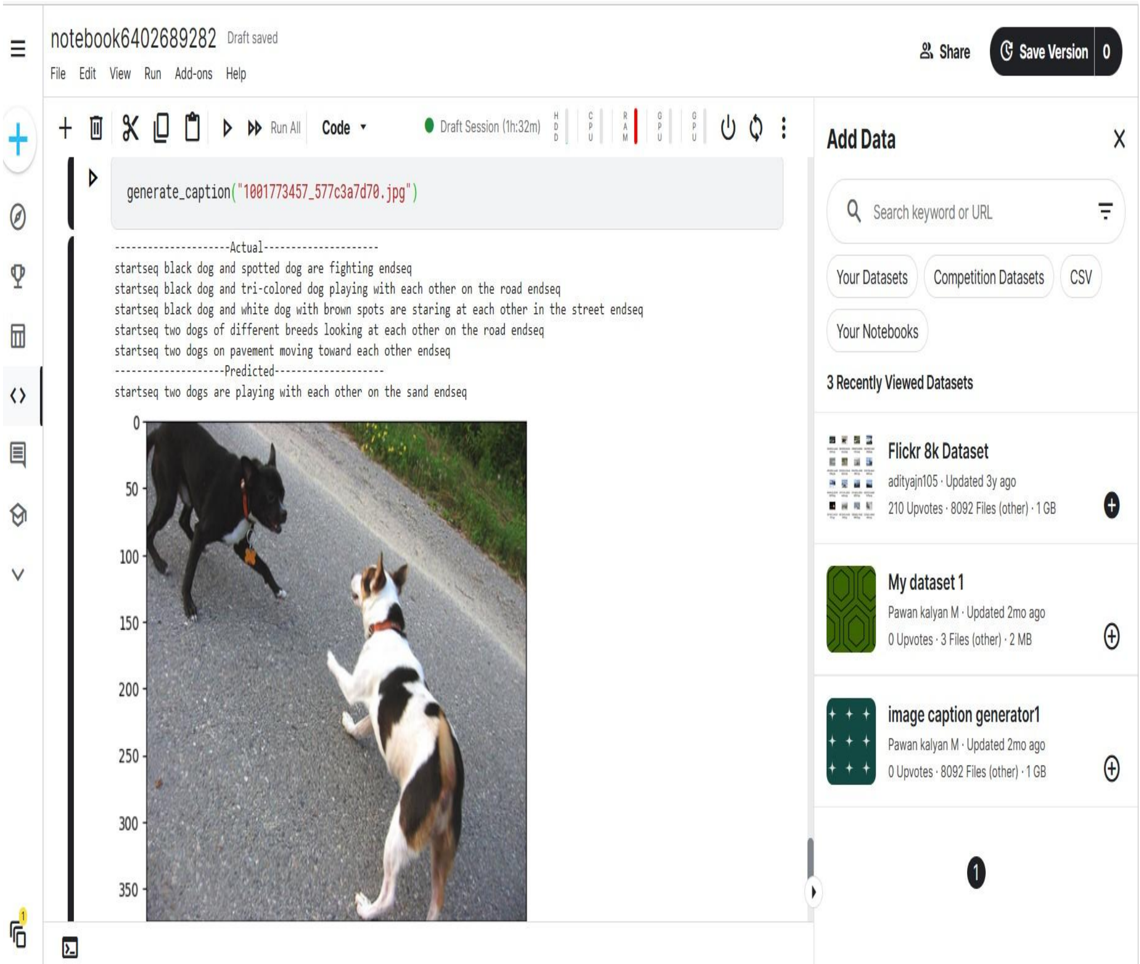


Screenshot 5.3: Pre -Processing Text Data

\



Screenshot 5.4: Model Getting Trained



Screenshot 5.5: Generation Of Caption To The Trained Model

6.TESTING

6. TESTING

6.1 INTRODUCTION TO TESTING

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

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

Individual software modules or components are reviewed during unit testing, which is a sort of software testing. The goal is to make sure that every line of software code works. Unit testing is done by developers throughout the application development (code) process. Unit tests are used to isolate and verify that code is accurate. A single function, method, process, module, or object is referred to as a unit.

In the SDLC, STLC, and V Models, unit testing is the initial level of testing performed before integration testing. Unit testing is a type of WhiteBox testing performed by programmers. In the actual world, QA engineers perform testing due to time constraints or developers' dislike of it.

6.2.2 Integration Testing:

Integration testing is used to ensure that modules/components perform as intended when they are combined, i.e. to ensure that modules that work well separately do not cause problems when combined. When it comes to testing large applications with the black box testing technique, it entails the use of a number of modules that are tightly associated with one another. We may use the concepts of integration testing to test these types of scenarios.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, orspecial test cases.

6.3 TEST CASES

6.3.1 CLASSIFICATION

Test case ID	Test case name	Purpose	Input	output
1	Gesture Processing Detection.	To detect Whether given image is real or fake.	The user gives the input in the form of image to the Trained Model.	An output is a text message for given image to the model.
2	Image Processing Detection.	To detect Whether given image is real or fake.	The user gives the input in the form of image to the Trained Model.	An output is a text message for given image to the model.

7. CONCLUSION

7. CONCLUSION & FUTURE SCOPE

7.1 PROJECT CONCLUSION

Different Deep learning methods for caption generation have been reviewed in this paper. Generating an appropriate and grammatically correct caption in a natural language like a human is a difficult task ,this task involves feature extraction and natural language processing concepts. the data set that has been used for this model is flickr_8k data set which consists around eight thousand images, for the given image the context of the image is recognized by targeting multiple objects. We have understood the architectures and wide ranging applications of CNN and LSTM .As the users are increasing day by day the image captioning has a vast scope in the future, the model is built over a dataset of eight thousand images to improve the accuracy and performance this can be implemented on the higher dataset.

7.2 Future enhancements

Image captioning is used in many applications including Google photos, Skin vision, Adobe photoshop, social media etc. It has recently become a significant issue, and we have explored several image generating strategies in the past, understand the process of image extractionand addressed the challenges faced by them, to improve the accuracy by extracting the features of the context of the image larger data set can be used. This project can be enhanced in the future by using CPUs and GPUs for achieving faster, cheaper and efficient algorithms, instead of limiting to the one language i.e , English the caption can be generated in different languages as required by the user.

8.BIBLIOGRAPHY

8. BIBLIOGRAPHY

8.1REFERENCES

[1] S. ALBAWI and T. A. MOHAMMED, "Understanding of a Convolutional Neural Network,"in ICET, Antalya, 2017.

[2] S. Hochreiter, "LONG SHORT-TERM MEMORY," Neural Computation, December 1997.

[3] O. Vinyals, A. Toshev, S. Bengio and D. Erhan,"A Neural Image Caption Generator," CVPR
2015 Open Access Repository, vol. Xiv, 17 November 2014.

[4] D. S. Whitehead, L. Huang, H. and S.-F. Chang, "Entityaware Image CaptionGeneration,"inEmpirical Methods in Natural Language Processing, Brussels, 2018.

[5] C. Elamri and T. Planque, "Automated Neural Image Caption Generator for Visually Impaired
People," California, 2016.

[6] G. Ding, M. Chen, S. Zhao, H. Chen, J. Han and Q. Liu, "Neural Image Caption Generation
with Weighted Training and Reference," Cognitive Computation, 08 August 2018.

[7] J. Chen, W. Dong and M. Li, "Image Caption Generator Based On Deep Neural Networks,"
March 2018. [8] S. Bai and S. An,"A Survey on Automatic Image Caption Generation,"
Neurocomputing, 13 April 2018. [9] R. Staniute and D. Sesok, "A Systematic Literature Review
on Image Captioning," Applied Sciences, vol. 9, no. 10, 16 March 2019.

[8] J. Hessel, N. Savva and M. J. Wilber, "Image Representations and New Domains in Neural
Image Captioning," ACL Anthology, vol. Proceedings of the Fourth Workshop on Vision and
Language, p. 29–39, 18 September 2015.

[9] M. Z. Hossain, F. SOHEL, M. F. SHIRATUDDIN and H. LAGA, "A Comprehensive

Survey of Deep Learning for Image Captioning," ACM Journals, vol. 51, no. 6, 14 Octomber 2018.

[10] A. Farhadi, . M. Hejrati, . M. A. Sadeghi and . P. Young, "Every Picture Tells a Story: Generating Sentences from Images," in ACM Digital Library, 2010. for Visual Recognition and Description," CVPR 2015, vol. 14, 31 May 2016.

8.2 GHITHUB LINK:

kawshikbhyroju/Major-Project (github.com)