GLA UNIVERSITY
ESTD. 2010
ऋते ज्ञानान्न मुक्तिः

GLA
UNIVERSITY
MATHURA
Recognised by UGC Under Section 2(f)

DEPARTMENT OF COMPUTER ENGINEERING & APPLICATIONS
INSTITUTE OF ENGINEERING & TECHNOLOGY

B.Tech. IV Year CSE

Project Report

On

"Image Caption Generator Using CNN & LSTM"

Under the supervision of

Mr. Kailash Kumar

Submitted by

1. Harshita Bhardwaj (F-28/ 181500261)
2. Priyanshu Upadhyay (D-51/ 181500517)
3. Umesh Pratap Singh (F-64/ 181500767)
4. Vipul (F-67/ 181500801)

Group No: G-16

Odd Semester, 2021-22

Department of computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,
Mathura – 281406

Declaration

We hereby declare that the work which is being presented in the Major Project I **"**Image Caption Generator Using CNN & LSTM**",** in partial full fillment of the requirements for Major Project viva voce, is an authentic record of our own work carried under the supervision of **Mr. Kailash Kumar, Assistant Professor, GLA University, Mathura.**

Harshita Bhardwaj (181500261)
Sign: _____________

Priyanshu Upadhyay (181500517)
Sign: _____________

Vipul (181500801)
Sign: _____________

Umesh Pratap Singh (181500767)
Sign: _____________

Course: B.Tech(CSE)
Year: 4rd
Semester: VII
Members GitHub Id's:

- https://github.com/harshitabhardwaj16
- https://github.com/vipulgupta22
- https://github.com/ priyanshudec23
- https://github.com/thakurups

Department of computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,
Mathura – 281406

Certificate

This is to certify that the project entitled "Image Caption Generator Using CNN & LSTM" carried out in Major Project I is the work done by Harshita Bhardwaj, Vipul, Priyanshu Upadhyay and Umesh Pratap Singh is submitted in partial full fillment of the requirements for the award of degree Bachelor of Technology (Computer Science and Engineering).

Signature of Supervisor:

Name of Supervisor: Mr. Kailash Kumar

Date:

Abstract

In this project, we use CNN and LSTM to identify the caption of the image. As the deep learning techniques are growing, huge datasets and computer power are helpful to build models that can generate captions for an image. This is what we are going to implement in this Python based project where we will use deep learning techniques like CNN and RNN. Image caption generator is a process which involves natural language processing and computer vision concepts to recognize the context of an image and present it in English. In this survey paper, we carefully follow some of the core concepts of image captioning and its common approaches. We discuss Keras library, numpy and jupyter notebooks for the making of this project. We also discuss about flickr_dataset and CNN used for image classification.

Table of Content

Chapter-1
Introduction

1.1. Overview & Motivation

Every day, we encounter a large number of images from various sources such as the internet, news articles, document diagrams and advertisements. These sources contain images that viewers would have to interpret themselves. Most images do not have a description, but the human can largely understand them without their detailed captions. However, machine needs to interpret some form of image captions if humans need automatic image captions from it.

Image captioning is important for many reasons. Captions for every image on the internet can lead to faster and descriptively accurate images searches and indexing.
Ever since researchers started working on object recognition in images, it became clear that only providing the names of the objects recognized does not make such a good impression as a full human-like description. As long as machines do not think, talk, and behave like humans, natural language descriptions will remain a challenge to be solved.

Image captioning has various applications in various fields such as biomedicine, commerce, web searching and military etc. Social media like Instagram , Facebook etc can generate captions automatically from images.

Generating captions for images is a vital task relevant to the area of both Computer Vision and Natural Language Processing. Mimicking the human ability of providing descriptions for images by a machine is itself a remarkable step along the line of Artificial Intelligence. The main challenge of this task is to capture how objects relate to each other in the image and to express them in a natural language (like English).Traditionally, computer systems have been using pre-defined templates for generating text descriptions for images. However, 1 this approach does not provide sufficient variety required for generating lexically rich text descriptions. This shortcoming has been suppressed with the increased efficiency of neural networks. Many state of art models use neural networks for generating captions by taking image as input and predicting next lexical unit in the output sentence.

1.2. Objective

The objective of our project is to learn the concepts of a CNN and LSTM model and build a working model of Image caption generator by implementing CNN with LSTM.

In this Python project, we will be implementing the caption generator using CNN (Convolutional Neural Networks) and LSTM (Long short term memory). The image features will be extracted from Xception which is a CNN model trained on the imagenet dataset and then we feed the features into the LSTM model which will be responsible for generating the image captions.

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

The main aim of this project is to get a little bit of knowledge of deep learning techniques. We use two techniques mainly CNN and LSTM for image classification. So, to make our image caption generator model, we will be merging these architectures. It is also called a CNN-RNN model.

CNN is used for extracting features from the image. We will use the pre-trained model Xception.

LSTM will use the information from CNN to help generate a description of the image.

1.3. Scope

Future work Image captioning has become an important problem in recent days due to the exponential growth of images in social media and the internet. This report discusses the various research in image retrieval used in the past and it also highlights the various techniques and methodology used in the research. As feature extraction and similarity calculation in images are challenging in this domain, there is a tremendous scope of possible research in the future. Current image retrieval systems use similarity calculation by making use of features such as color, tags, IMAGE RETRIEVAL USING IMAGE CAPTIONING 54 histogram, etc. There cannot be completely accurate results as these methodologies do not depend on the context of the image. Hence, a complete research in image retrieval making use of context of the images such as image captioning will facilitate to solve this problem in the future. This project can be further enhanced in future to improve the identification of classes which has a lower precision by training it with more image captioning datasets. This methodology can also be combined with previous image retrieval methods such as histogram, shapes, etc. and can be checked if the image retrieval results get better.

Chapter-2
Literature Survey

The image captioning problem and its proposed solutions have existed since the advent of the Internet and its widespread adoption as a medium to share images. Numerous algorithms and techniques have been put forward by researchers from different perspectives. Krizhevsky et al.

Implemented a neural network using non-saturating neurons and a very efficient a unique method GPU implementation of the convolution function.[1] By employing a regularization method called dropout, they succeeded in reducing overfitting. Their neural network consisted of maxpooling layers and a final 1000-way softmax. Deng et al.

Introduced a new database which they called ImageNet, an extensive collection of images built using the core of the WordNet structure[2]. ImageNet organized the different classes of images in a densely populated semantic hierarchy. Karpathy and FeiFei

Made use of datasets of images and their sentence descriptions to learn about the inner correspondences visual data and language.[3] Their work described a Multimodal Recurrent Neural Network architecture that utilises the inferred co-linear arrangement of features in order to learn how to generate novel descriptions of images. Yang et al.

Proposed a system for the automatic generation of a natural language description of an image, which will help immensely in furthering image understanding.[4] The proposed multimodel neural network method, consisting of object detection and localization modules, is very similar to the human visual system which is able to learns how to describe the content of images automatically. In order to address the problem of LSTM units being complex and inherently sequential across time, Aneja et al.

Proposed a convolutional [5] network model for machine translation and conditional image generation. Pan et. al

Experimented extensively with multiple network architectures on large datasets consisting of varying content styles,[6] and proposed a unique model showing noteworthy improvement on captioning accuracy over the previously proposed models. Vinyals et al.

Presented a generative model consisting of a deep recurrent architecture that leverages machine translation and computer vision, [7] used to generate natural descriptions of an image by ensuring highest probability of the generated sentence to accurately describe the target image. Xu et al.

Introduced an attention based model that learned to describe the image regions automatically. The model was trained using standard backpropagation techniques by maximizing a variable lower bound [8]. The model was able to automatically learn identify object boundaries while at the same time generate an accurate descriptive sentence.

The problem of generating natural language descriptions from visual data has long been studied in computer vision, but mainly for video. [9] This has led to complex systems composed of visual primitive recognizers combined with a structured formal language, e.g. And-Or Graphs or logic systems, which are further converted to natural language via rule-based systems. Such systems are heavily hand-designed, relatively brittle and have been demonstrated only on limited domains, e.g. traffic scenes or sports.

The problem of still image description with natural text has gained interest more recently. Leveraging recent advances in recognition of objects, their attributes and locations, allows us to drive natural language generation systems, though these are limited in their expressivity. [10] Farhadi et al. use detections to infer a triplet of scene elements which is converted to text using templates. Similarly, Li et al. start off with detections and piece together a final description using phrases containing detected objects and relationships.[11] A more complex graph of detections beyond triplets is used by Kulkani et al. but with template-based text generation. More powerful language models based on language parsing have been used as well. The above approaches have been able to describe images “in the wild”, but they are heavily hand designed and rigid when it comes to text generation.

Chapter-3
Proposed Model

The main aim of this project is to get a little bit of knowledge of deep learning techniques. We use two techniques mainly CNN and LSTM for image classification. So, to make our image caption generator model, we will be merging these architectures. It is also called a CNN-RNN model.

CNN is used for extracting features from the image. We will use the pre-trained model Xception.

LSTM will use the information from CNN to help generate a description of the image.

3.1 Convolutional Neural Network

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.The pre-processing required in a ConvNet is much lower as compared to other classification algorithms[1].

Convolutional Neural networks are specialized deep neural networks which can process the data that has input shape like a 2D matrix. Images are easily represented as a 2D matrix and CNN is very useful in working with images.

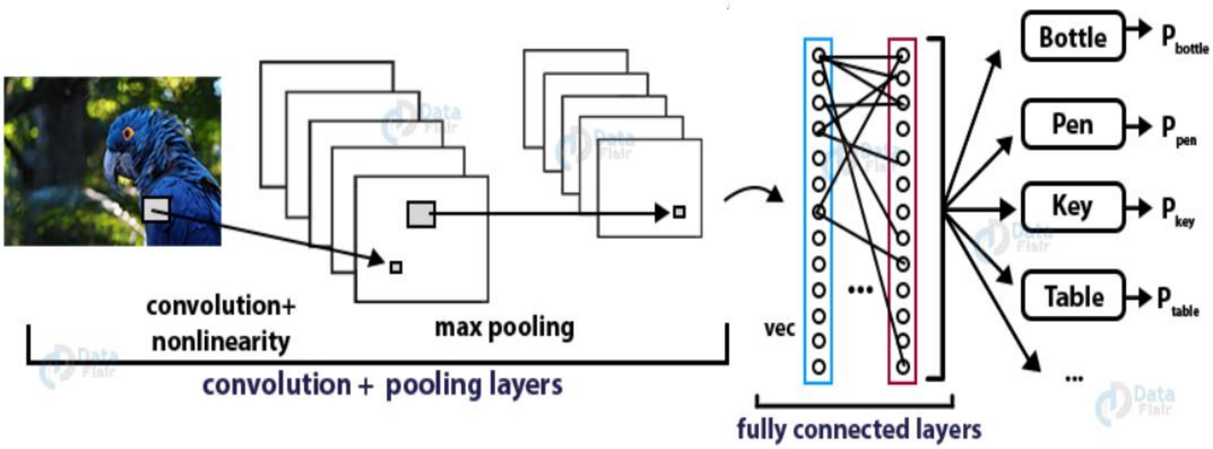


Figure-1 Working of CNN

It scans images from left to right and top to bottom to pull out important features from the image and combines the feature to classify images. It can handle the images that have been translated, rotated, scaled and changes in perspective.

3.2 Long Short Term Memory

LSTM stands for Long short term memory, they are a type of RNN (recurrent neural network) which is well suited for sequence prediction problems. Based on the previous text, we can predict what the next word will be. It has proven itself effective from the traditional RNN by overcoming the limitations of RNN which had short term memory. LSTM can carry out relevant information throughout the processing of inputs and with a forget gate, it discards non-relevant information.

LSTMs are designed to overcome the vanishing gradient problem and allow them to retain information for longer periods compared to traditional RNNs. LSTMs can maintain a constant error, which allows them to continue learning over numerous time-steps and backpropagate through time and layers[2].

LSTMs use gated cells to store information outside the regular flow of the RNN. With these cells, the network can manipulate the information in many ways, including storing information in the cells and reading from them. The cells are individually capable of making decisions regarding the information and can execute these decisions by opening or closing the gates. The ability to retain information for a long period of time gives LSTM the edge over traditional RNNs in these tasks.

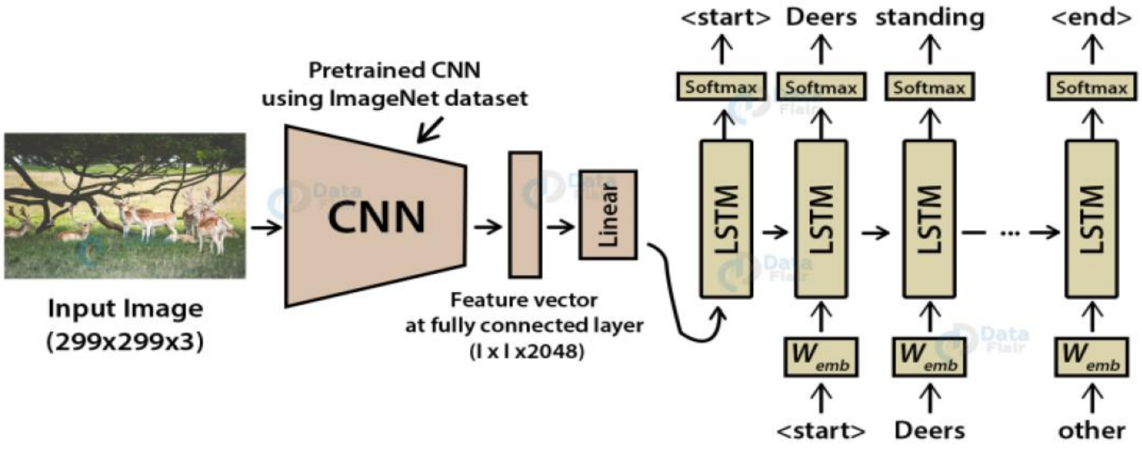


Figure-2 Model, Image Caption Generator

The chain-like architecture of LSTM allows it to contain information for longer time periods, solving challenging tasks that traditional RNNs struggle to or simply cannot solve.

The three major parts of the LSTM include:

Forget gate—removes information that is no longer necessary for the completion of the task. This step is essential to optimizing the performance of the network.

Input gate—responsible for adding information to the cells

Output gate—selects and outputs necessary information

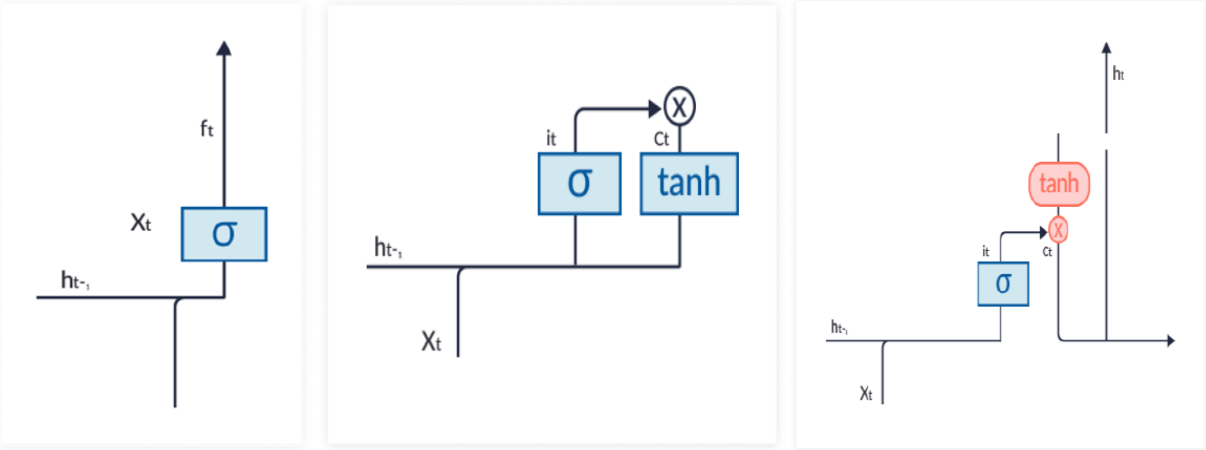


Figure-3 Forget Gate, Input Gate, Output Gate

The CNN LSTM architecture involves using Convolutional Neural Network (CNN) layers for feature extraction on input data combined with LSTMs to support sequence prediction[3]. This architecture was originally referred to as a Long-term Recurrent Convolutional Network or LRCN model, although we will use the more generic name “CNN LSTM” to refer to LSTMs that use a CNN as a front end in this lesson.

This architecture is used for the task of generating textual descriptions of images. Key is the use of a CNN that is pre-trained on a challenging image classification task that is re-purposed as a feature extractor for the caption generating problem.

Chapter-4
Wireframe

4.1 Dataset

This project requires a dataset which have both images and their caption. The dataset should be able to train the image captioning model.

FLICKR8K DATASET

Flickr8k dataset is a public benchmark dataset for image to sentence description. This dataset consists of 8000 images with five captions for each image. These images are extracted from diverse groups in Flickr website. Each caption provides a clear description of entities and events present in the image. The dataset depicts a variety of events and scenarios and doesn‟t include images containing well-known people and places which makes the dataset more generic. The dataset has 6000 images in training dataset, 1000 images in development dataset and 1000 images in test dataset. Features of the dataset making it suitable for this project are: • Multiple captions mapped for a single image makes the model generic and avoids overfitting of the model. • Diverse category of training images can make the image captioning model to work for multiple categories of images and hence can make the model more robust.

4.2 Image Data Preparation

The image should be converted to suitable features so that they can be trained into a deep learning model. Feature extraction is a mandatory step to train any image in deep learning model. The features are extracted using Convolutional Neural Network (CNN) with Visual Geometry Group (VGG-16) model. This model also won ImageNet Large Scale Visual Recognition Challenge in 2015 to classify the images into one among the 1000 classes given in the challenge[4]. Hence, this model is ideal to use for this project as image captioning requires identification of images. In VGG-16, there are 16 weight layers in the network and the deeper number of layers help in better feature extraction from images. The VGG-16 network uses 3*3 convolutional layers making its architecture simple and uses max pooling layer in between to reduce volume size of the image. The last layer of the image which predicts the classification is removed and the internal representation of image just before classification is returned as feature. The dimension of the input image should be 224*224 and this model extracts features of the image and returns a 1-dimensional 4096 element vector.

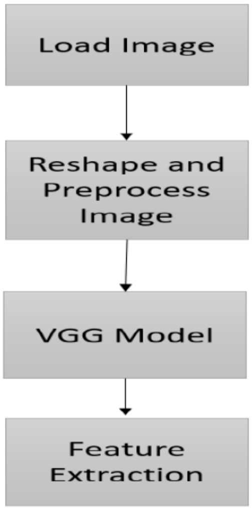


Figure-3 Feature Extraction in images using VGG

4.3 Data Cleaning

Flickr8k dataset contains multiple descriptions described for a single image. In the data preparation phase, each image id is taken as key and its corresponding captions are stored as values in a dictionary.

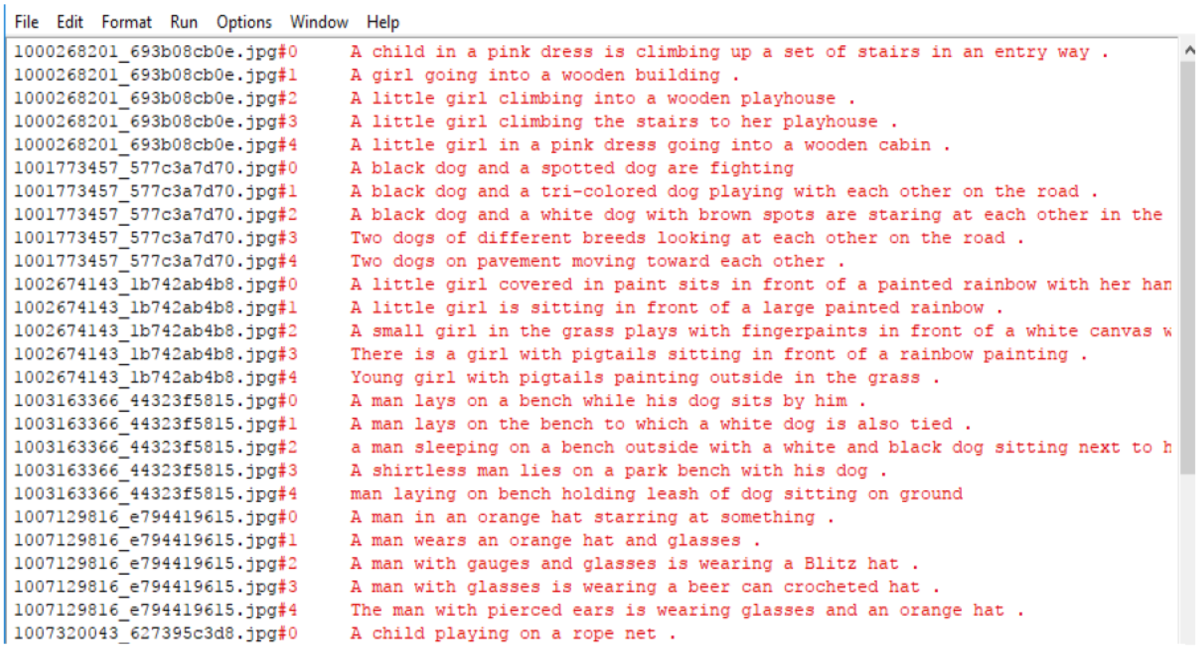
In order to make the text dataset work in machine learning or deep learning models, raw text should be converted to a usable format[5]. The following text cleaning steps are done before using it for the project: • Removal of punctuations. • Removal of numbers. • Removal of single length words. • Conversion of uppercase to lowercase characters. Stop words are not removed from the text data as it will hinder the generation of a grammatically complete caption which is needed for this project. Table 1 shows samples of captions after data cleaning.

Original Captions	Captions after Data Cleaning
Two people are at the edge of a lake, facing the water and the city skyline.	two people are at the edge of lake facing the water and the city skyline
A little girl rides in a child 's swing.	little girl rides in child swing
Two boys posing in blue shirts and khaki shorts.	two boys posing in blue shirts and khaki shorts

Table-1 Data cleaning of captions

4.3.1 Getting and Performing Data Cleaning

The main text file which contains all image captions is **Flickr8k.token** in our **Flickr_8k_text** folder.



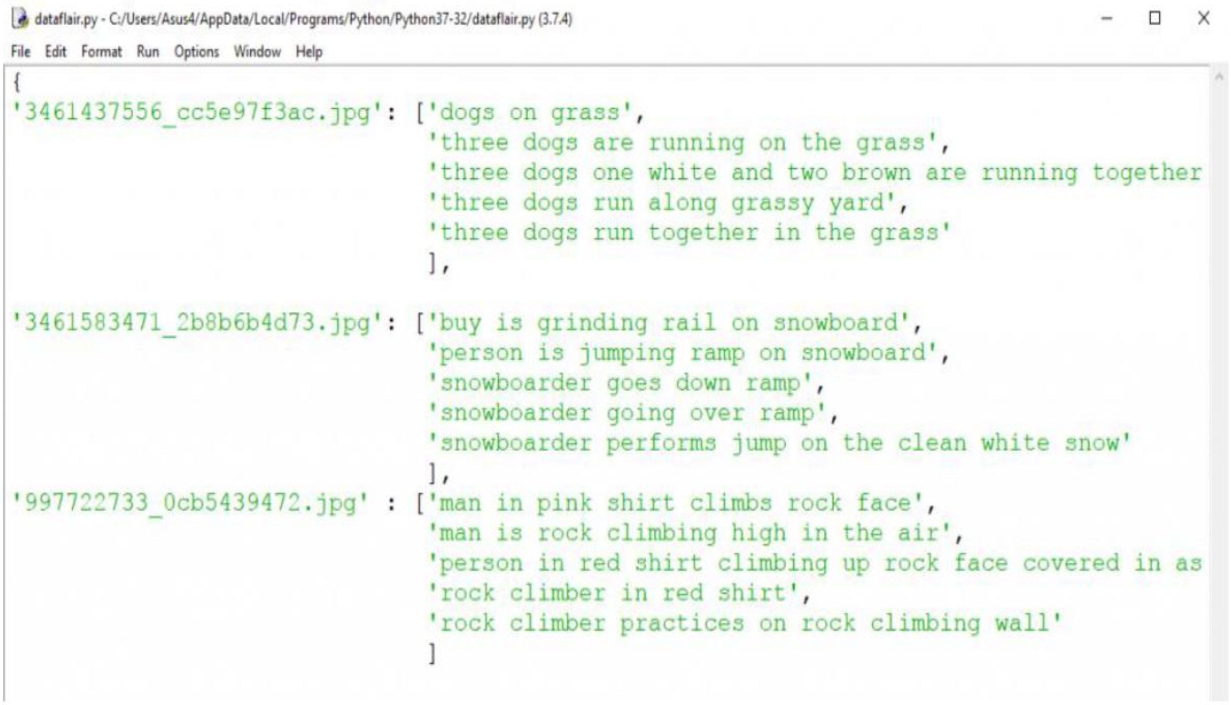
```
1000268201_693b08cb0e.jpg#0	A child in a pink dress is climbing up a set of stairs in an entry way .
1000268201_693b08cb0e.jpg#1	A girl going into a wooden building .
1000268201_693b08cb0e.jpg#2	A little girl climbing into a wooden playhouse .
1000268201_693b08cb0e.jpg#3	A little girl climbing the stairs to her playhouse .
1000268201_693b08cb0e.jpg#4	A little girl in a pink dress going into a wooden cabin .
1001773457_577c3a7d70.jpg#0	A black dog and a spotted dog are fighting
1001773457_577c3a7d70.jpg#1	A black dog and a tri-colored dog playing with each other on the road .
1001773457_577c3a7d70.jpg#2	A black dog and a white dog with brown spots are staring at each other in the
1001773457_577c3a7d70.jpg#3	Two dogs of different breeds looking at each other on the road .
1001773457_577c3a7d70.jpg#4	Two dogs on pavement moving toward each other .
1002674143_1b742ab4b8.jpg#0	A little girl covered in paint sits in front of a painted rainbow with her han
1002674143_1b742ab4b8.jpg#1	A little girl is sitting in front of a large painted rainbow .
1002674143_1b742ab4b8.jpg#2	A small girl in the grass plays with fingerpaints in front of a white canvas w
1002674143_1b742ab4b8.jpg#3	There is a girl with pigtails sitting in front of a rainbow painting .
1002674143_1b742ab4b8.jpg#4	Young girl with pigtails painting outside in the grass .
1003163366_44323f5815.jpg#0	A man lays on a bench while his dog sits by him .
1003163366_44323f5815.jpg#1	A man lays on the bench to which a white dog is also tied .
1003163366_44323f5815.jpg#2	a man sleeping on a bench outside with a white and black dog sitting next to h
1003163366_44323f5815.jpg#3	A shirtless man lies on a park bench with his dog .
1003163366_44323f5815.jpg#4	man laying on bench holding leash of dog sitting on ground
1007129816_e794419615.jpg#0	A man in an orange hat starring at something .
1007129816_e794419615.jpg#1	A man wears an orange hat and glasses .
1007129816_e794419615.jpg#2	A man with gauges and glasses is wearing a Blitz hat .
1007129816_e794419615.jpg#3	A man with glasses is wearing a beer can crocheted hat .
1007129816_e794419615.jpg#4	The man with pierced ears is wearing glasses and an orange hat .
1007320043_627395c3d8.jpg#0	A child playing on a rope net .
```

Figure-4 Flicker DataSet text format

The format of our file is image and caption separated by a new line (“\n”).

Each image has 5 captions and we can see that #(0 to 5)number is assigned for each caption. We will define 5 functions:

- **load_doc(filename) –** For loading the document file and reading the contents inside the file into a string.
- **all_img_captions(filename) –** This function will create a **descriptions** dictionary that maps images with a list of 5 captions. The descriptions dictionary will look something like the Figure.


dataflair.py - C:/Users/Asus4/AppData/Local/Programs/Python/Python37-32/dataflair.py (3.7.4)
File Edit Format Run Options Window Help

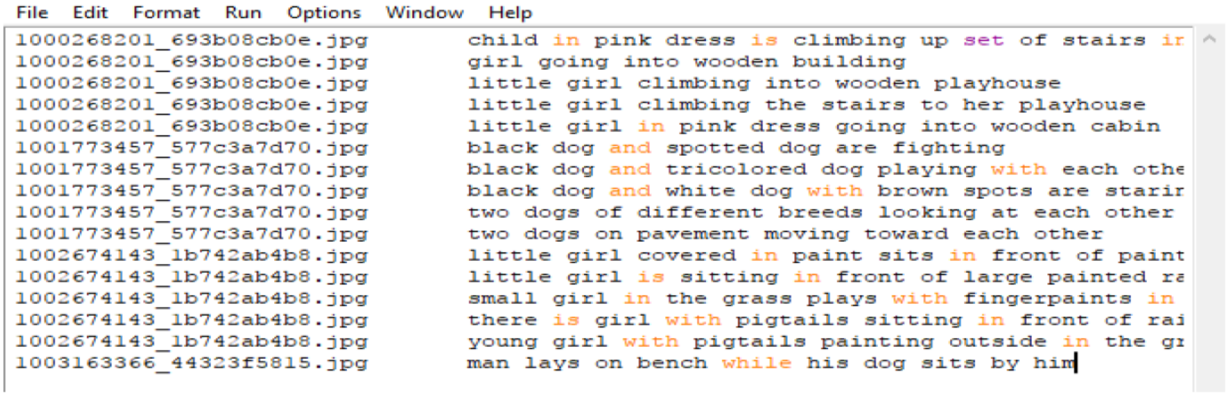
```
{
'3461437556_cc5e97f3ac.jpg': ['dogs on grass',
                              'three dogs are running on the grass',
                              'three dogs one white and two brown are running together
                              'three dogs run along grassy yard',
                              'three dogs run together in the grass'
                              ],

'3461583471_2b8b6b4d73.jpg': ['buy is grinding rail on snowboard',
                              'person is jumping ramp on snowboard',
                              'snowboarder goes down ramp',
                              'snowboarder going over ramp',
                              'snowboarder performs jump on the clean white snow'
                              ],
'997722733_0cb5439472.jpg' : ['man in pink shirt climbs rock face',
                              'man is rock climbing high in the air',
                              'person in red shirt climbing up rock face covered in as
                              'rock climber in red shirt',
                              'rock climber practices on rock climbing wall'
                              ]
```

Figure-5 Flicker Dataset Python File

- **cleaning_text(descriptions) –** This function takes all descriptions and performs data cleaning. This is an important step when we work with textual data, according to our goal, we decide what type of cleaning we want to perform on the text[6]. In our case, we will be removing punctuations, converting all text to lowercase and removing words that contain numbers.So, a caption like “A man riding on a three-wheeled wheelchair” will be transformed into “man riding on three wheeled wheelchair”

- **text_vocabulary(descriptions) –** This is a simple function that will separate all the unique words and create the vocabulary from all the descriptions.

- **save_descriptions(descriptions, filename) –** This function will create a list of all the descriptions that have been preprocessed and store them into a file. We will create a descriptions.txt file to store all the captions. It will look something like this:

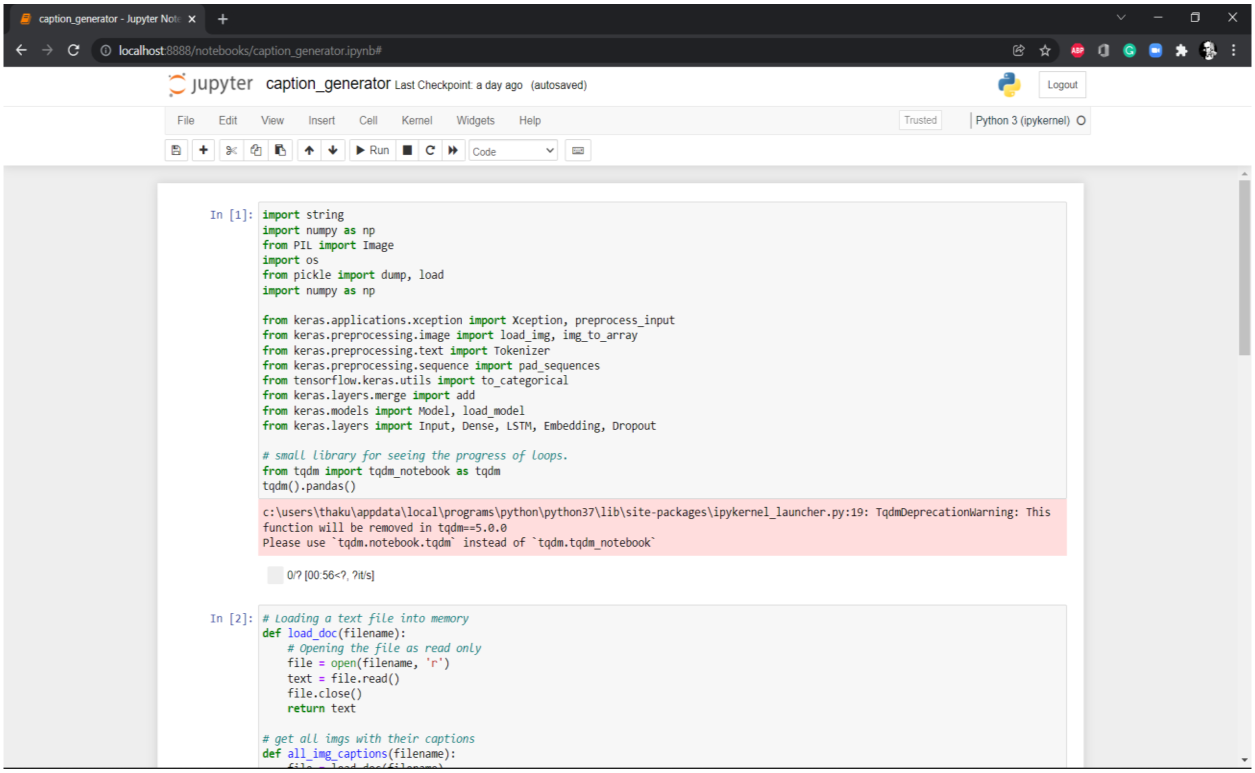


```
1000268201_693b08cb0e.jpg        child in pink dress is climbing up set of stairs ir
1000268201_693b08cb0e.jpg        girl going into wooden building
1000268201_693b08cb0e.jpg        little girl climbing into wooden playhouse
1000268201_693b08cb0e.jpg        little girl climbing the stairs to her playhouse
1000268201_693b08cb0e.jpg        little girl in pink dress going into wooden cabin
1001773457_577c3a7d70.jpg        black dog and spotted dog are fighting
1001773457_577c3a7d70.jpg        black dog and tricolored dog playing with each othe
1001773457_577c3a7d70.jpg        black dog and white dog with brown spots are starir
1001773457_577c3a7d70.jpg        two dogs of different breeds looking at each other
1001773457_577c3a7d70.jpg        two dogs on pavement moving toward each other
1002674143_1b742ab4b8.jpg        little girl covered in paint sits in front of paint
1002674143_1b742ab4b8.jpg        little girl is sitting in front of large painted ra
1002674143_1b742ab4b8.jpg        small girl in the grass plays with fingerpaints in
1002674143_1b742ab4b8.jpg        there is girl with pigtails sitting in front of rai
1002674143_1b742ab4b8.jpg        young girl with pigtails painting outside in the gr
1003163366_44323f5815.jpg        man lays on bench while his dog sits by him
```

Figure-6 Description of Images

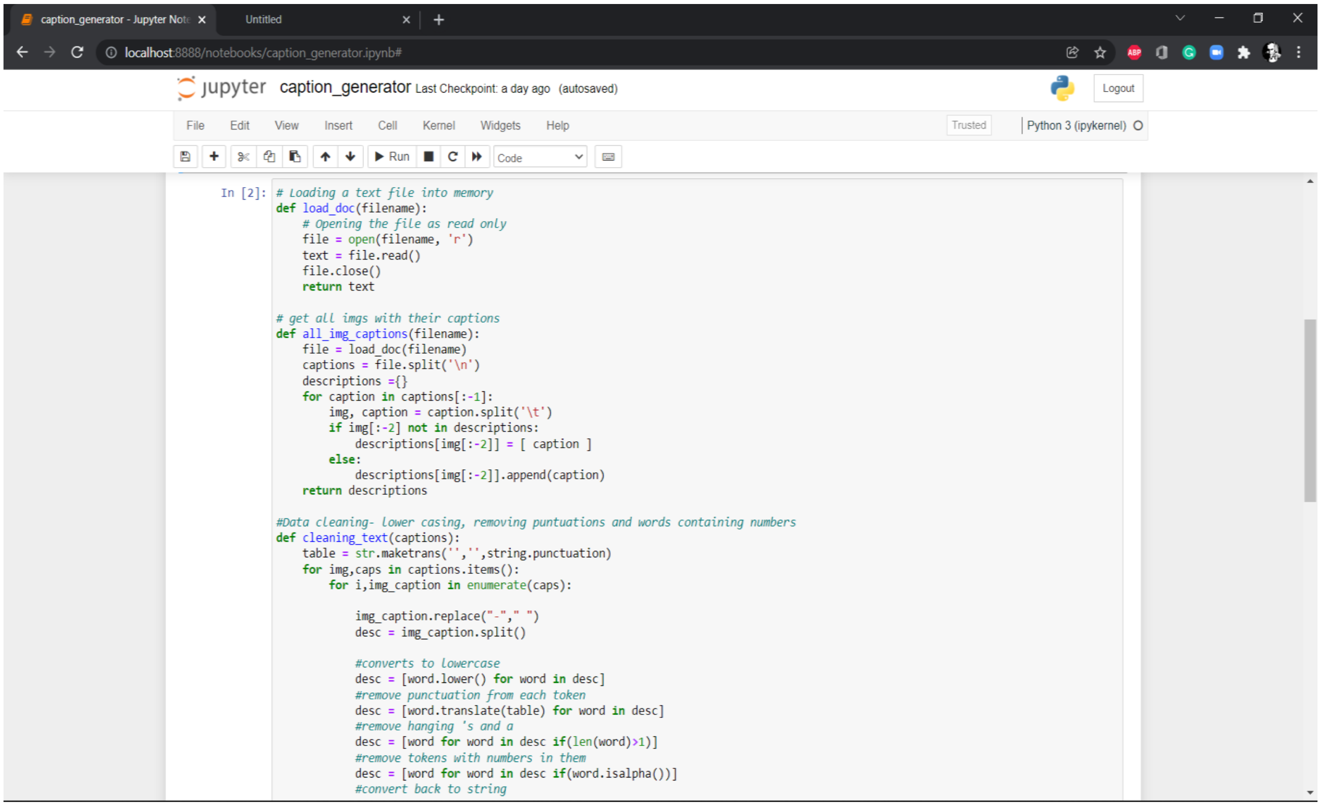
4.4 Implementation

4.4.1 Importing Libraries

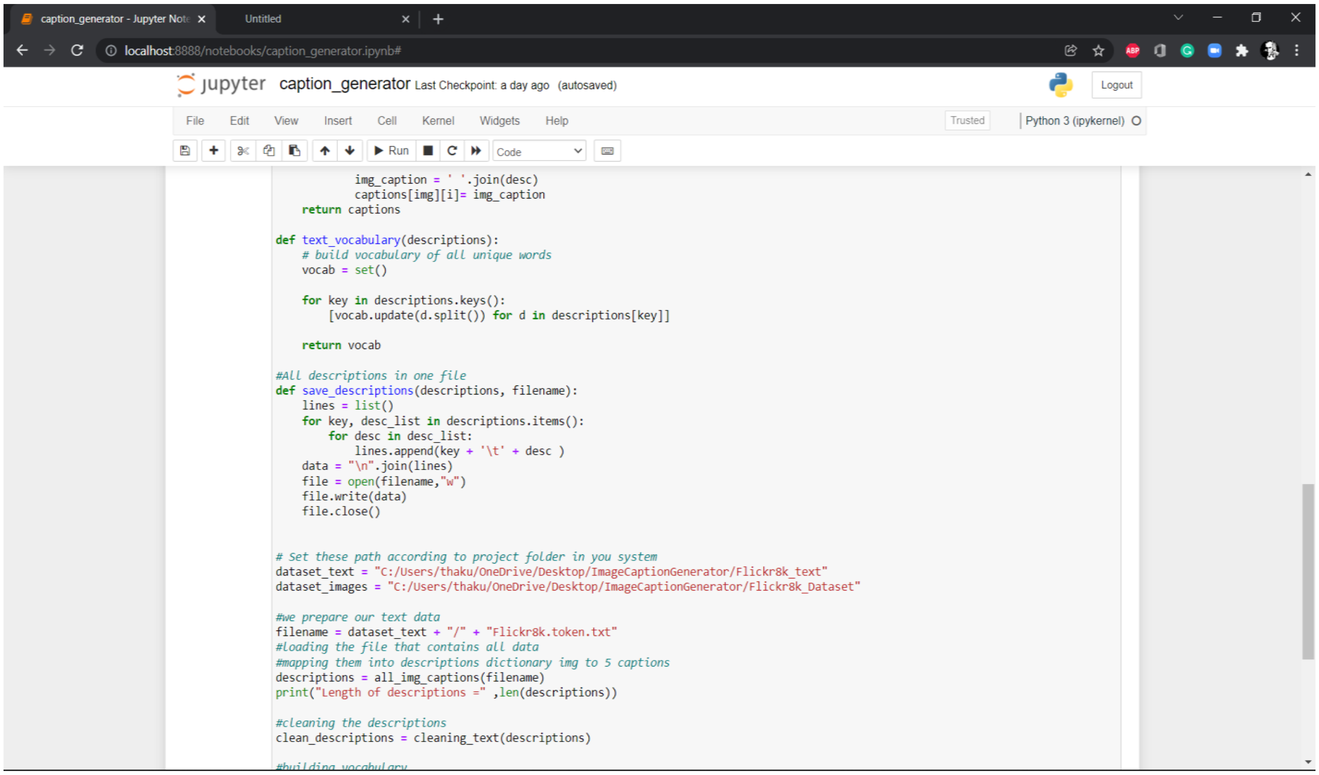

caption_generator - Jupyter Note
localhost:8888/notebooks/caption_generator.ipynb#
jupyter caption_generator Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [1]: import string
        import numpy as np
        from PIL import Image
        import os
        from pickle import dump, load
        import numpy as np

        from keras.applications.xception import Xception, preprocess_input
        from keras.preprocessing.image import load_img, img_to_array
        from keras.preprocessing.text import Tokenizer
        from keras.preprocessing.sequence import pad_sequences
        from tensorflow.keras.utils import to_categorical
        from keras.layers.merge import add
        from keras.models import Model, load_model
        from keras.layers import Input, Dense, LSTM, Embedding, Dropout

        # small library for seeing the progress of loops.
        from tqdm import tqdm_notebook as tqdm
        tqdm().pandas()

        c:\users\thaku\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:19: TqdmDeprecationWarning: This
        function will be removed in tqdm==5.0.0
        Please use `tqdm.notebook.tqdm` instead of `tqdm.tqdm_notebook`

        0/? [00:56<?, ?it/s]

In [2]: # Loading a text file into memory
        def load_doc(filename):
            # Opening the file as read only
            file = open(filename, 'r')
            text = file.read()
            file.close()
            return text

        # get all imgs with their captions
        def all_img_captions(filename):
            file = load_doc(filename)
```

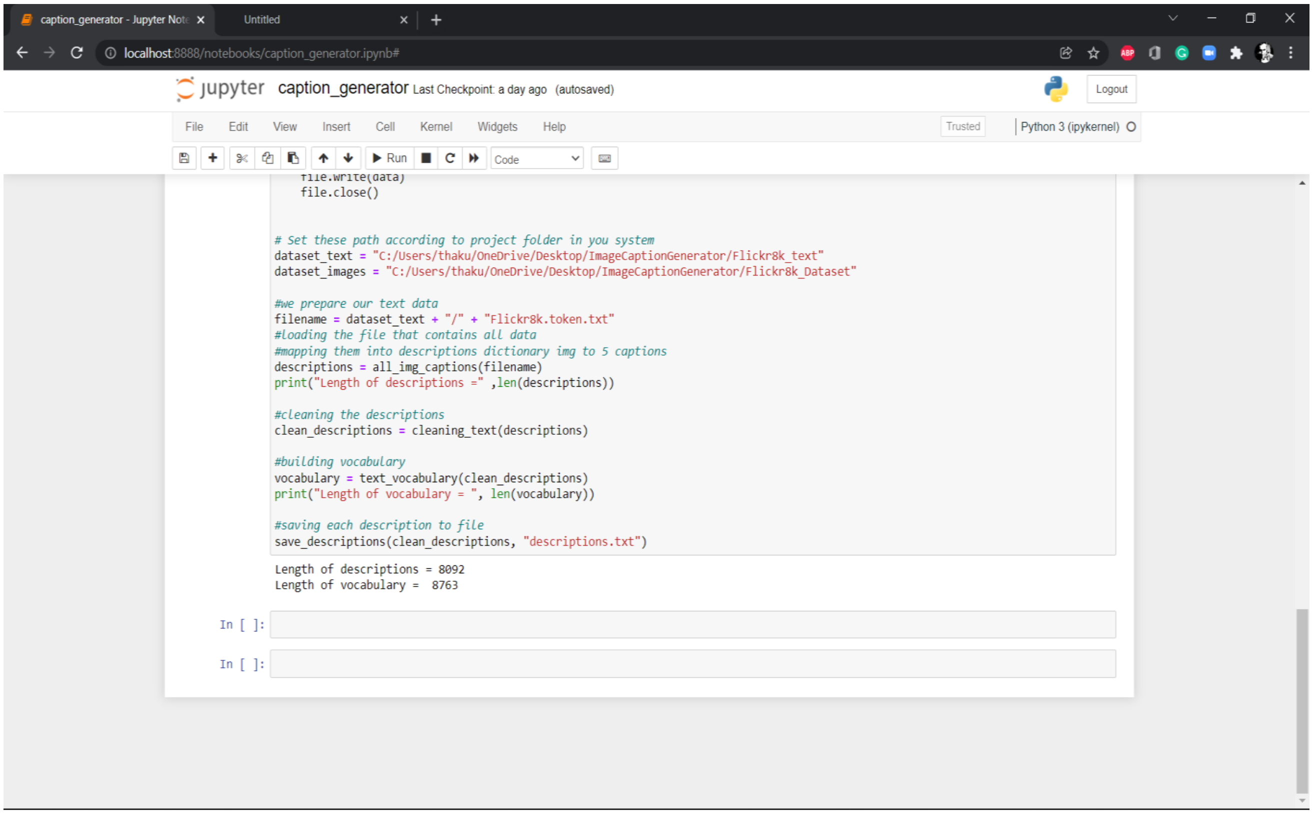
4.4.2 Data Cleaning


caption_generator - Jupyter Note
Untitled
localhost:8888/notebooks/caption_generator.ipynb#
jupyter
caption_generator Last Checkpoint: a day ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [2]: # Loading a text file into memory
        def load_doc(filename):
            # Opening the file as read only
            file = open(filename, 'r')
            text = file.read()
            file.close()
            return text

        # get all imgs with their captions
        def all_img_captions(filename):
            file = load_doc(filename)
            captions = file.split('\n')
            descriptions ={}
            for caption in captions[:-1]:
                img, caption = caption.split('\t')
                if img[:-2] not in descriptions:
                    descriptions[img[:-2]] = [ caption ]
                else:
                    descriptions[img[:-2]].append(caption)
            return descriptions

        #Data cleaning- lower casing, removing puntuations and words containing numbers
        def cleaning_text(captions):
            table = str.maketrans('','',string.punctuation)
            for img,caps in captions.items():
                for i,img_caption in enumerate(caps):

                    img_caption.replace("-"," ")
                    desc = img_caption.split()

                    #converts to lowercase
                    desc = [word.lower() for word in desc]
                    #remove punctuation from each token
                    desc = [word.translate(table) for word in desc]
                    #remove hanging 's and a
                    desc = [word for word in desc if(len(word)>1)]
                    #remove tokens with numbers in them
                    desc = [word for word in desc if(word.isalpha())]
                    #convert back to string
```


caption_generator - Jupyter Note
Untitled
localhost:8888/notebooks/caption_generator.ipynb#
jupyter
caption_generator Last Checkpoint: a day ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code

```
                    img_caption = ' '.join(desc)
                    captions[img][i]= img_caption
            return captions

        def text_vocabulary(descriptions):
            # build vocabulary of all unique words
            vocab = set()

            for key in descriptions.keys():
                [vocab.update(d.split()) for d in descriptions[key]]

            return vocab

        #All descriptions in one file
        def save_descriptions(descriptions, filename):
            lines = list()
            for key, desc_list in descriptions.items():
                for desc in desc_list:
                    lines.append(key + '\t' + desc )
            data = "\n".join(lines)
            file = open(filename,"w")
            file.write(data)
            file.close()


        # Set these path according to project folder in you system
        dataset_text = "C:/Users/thaku/OneDrive/Desktop/ImageCaptionGenerator/Flickr8k_text"
        dataset_images = "C:/Users/thaku/OneDrive/Desktop/ImageCaptionGenerator/Flickr8k_Dataset"

        #we prepare our text data
        filename = dataset_text + "/" + "Flickr8k.token.txt"
        #loading the file that contains all data
        #mapping them into descriptions dictionary img to 5 captions
        descriptions = all_img_captions(filename)
        print("Length of descriptions =" ,len(descriptions))

        #cleaning the descriptions
        clean_descriptions = cleaning_text(descriptions)

        #building vocabulary
```


caption_generator - Jupyter Note
Untitled
localhost:8888/notebooks/caption_generator.ipynb#
jupyter caption_generator Last Checkpoint: a day ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code

```
    file.write(data)
    file.close()


# Set these path according to project folder in you system
dataset_text = "C:/Users/thaku/OneDrive/Desktop/ImageCaptionGenerator/Flickr8k_text"
dataset_images = "C:/Users/thaku/OneDrive/Desktop/ImageCaptionGenerator/Flickr8k_Dataset"

#we prepare our text data
filename = dataset_text + "/" + "Flickr8k.token.txt"
#loading the file that contains all data
#mapping them into descriptions dictionary img to 5 captions
descriptions = all_img_captions(filename)
print("Length of descriptions =" ,len(descriptions))

#cleaning the descriptions
clean_descriptions = cleaning_text(descriptions)

#building vocabulary
vocabulary = text_vocabulary(clean_descriptions)
print("Length of vocabulary = ", len(vocabulary))

#saving each description to file
save_descriptions(clean_descriptions, "descriptions.txt")
```

```
Length of descriptions = 8092
Length of vocabulary =  8763
```

```
In [ ]:

In [ ]:
```

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, [Online] Available: https://papers.nips.cc/paper/4824- imagenetclassificationwith-deep-convolutionalneural-networks.pdf

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li and Li Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database

[3] Andrej Karpathy, Li Fei-Fei, Deep VisualSemantic Alignments for Generating Image Descriptions, [Online] Available: https://cs.stanford.edu/people/karpathy/cvpr2015.pdf

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

[5] Jyoti Aneja, Aditya Deshpande, Alexander Schwing, Convolutional Image Captioning, [Online] Available: https://arxiv.org/pdf/1711.09151.pdf

[6] Jia-Yu Pan, Hyung-Jeong Yang, Pinar Duygulu, Automatic Image Captioning, Conference: Conference: Multimedia and Expo, 2004. ICME ’04. 2004 IEEE International Conference on, Volume: 3

[7] Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan, Show and Tell: A Neural Image Caption Generator, [Online] Available: https://arxiv.org/pdf/1411.4555.pdf

[8] Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, Yoshua Bengio, Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, [Online] Available: https://arxiv.org/pdf/1502.03044.pdf [9] M. Hodosh, P. Young and J. Hockenmaier (2013) ”Framing Image Description as a Ranking Task: Data, Models and Evaluation Metrics”, Journal of Artificial Intelligence Research, Volume 47, pages 853-899

[9] R. Gerber and H.-H. Nagel. Knowledge representation for the generation of quantified natural language descriptions of vehicle traffic in image sequences. In ICIP. IEEE, 1996

[10] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences from images. In ECCV, 2010

[11] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg. Baby talk: Understanding and generating simple image descriptions. In CVPR, 2011

[12] Abhaya Agarwal and Alon Lavie. 2008. Meteor, m-bleu and m-ter: Evaluation metrics for high-correlation with human rankings of machine translation output. In Proceedings of the ThirdWorkshop on Statistical Machine Translation. Association for Computational Linguistics, 115–118.

[13] Ahmet Aker and Robert Gaizauskas. 2010. Generating image descriptions using dependency relational patterns. In Proceedings of the 48th annual meeting of the association for computational linguistics. Association for Computational Linguistics, 1250–1258.

[14] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In European Conference on Computer Vision. Springer, 382–398.

[15] 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 and vqa. arXiv preprint arXiv:1707.07998 (2017).