

# LOVELY PROFESSIONAL UNIVERSITY

## **“ADVANCED MACHINE LEARNING”**

*“PROJECT: IMAGE CAPTION GENERATOR”*

NAME: C. Ravisankar

REG: 11806888

ROLL: 25

SECTION: KM073

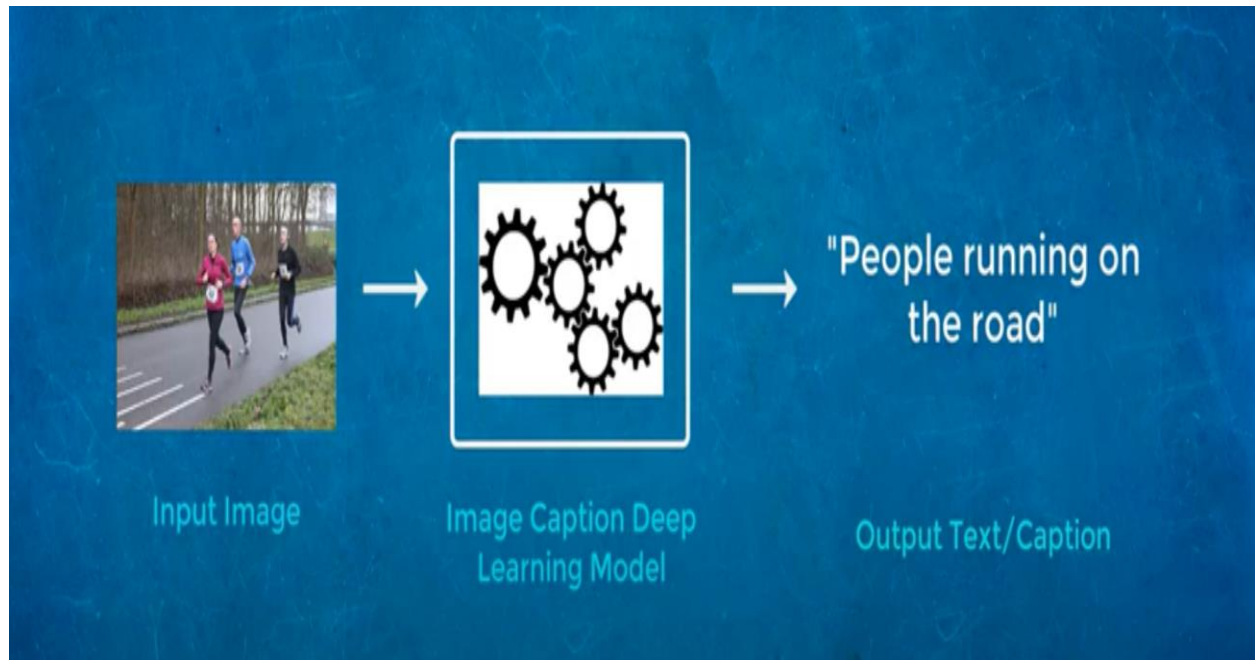
GITHUB LINK: <https://github.com/ravi2248/ML.git>

SUBMITTED TO

*ANKITA Ma'am*

## PROJECT NAME:

## IMAGE CAPTION GENERATOR



# **CONTENTS:**

- 1. INTRODUCTION**
- 2. LITERATURE REVIEW**
- 3. DATASET USED**
- 4. PROPOSED ARCHITECTURE**
- 5. RESULT AND EXPERIMENT ANALYSIS**
- 6. SCREENSHOTS**
- 7. CONCLUSION AND FUTURE SCOPE AND REFERENCES**

# INTRODUCTION

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

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

The biggest challenges is most definitely being able to create a description that must capture not only the objects contained in an image but also express how these objects relate to each other.

Consider the following image from the Flickr8k dataset:



What do we see in the above image?

We can easily say ‘A black dog and a brown dog in the snow’ or ‘The small dogs play in the snow’ or ‘Two Pomeranian dogs playing in the snow’. It seems easy for us as humans to look at an image like that and describe it appropriately.

## LITERATURE REVIEW

I get the details of the Image Caption Generator from the research papers. In which, I get to know the information and procedure of this project. These are the research papers which I take as reference of my project.

1. <https://cs.stanford.edu/people/karpathy/cvpr2015.pdf>
2. [https://www.ripublication.com/ijaer18/ijaerv13n9\\_102.pdf](https://www.ripublication.com/ijaer18/ijaerv13n9_102.pdf)
3. <https://arxiv.org/pdf/1411.4555.pdf>

# DATASET USED

Several datasets are used for training, testing, and evaluation of the image captioning methods. The datasets differ in various perspectives such as the number of images, the number of captions per image, format of the captions, and image size. Three datasets: Flickr8k, Flickr30k, and MS COCO Dataset are popularly used.

In the Flickr8k dataset, each image is associated with five different captions that describe the entities and the events depicted in the image that were collected. By associating each image with multiple, independently produced sentences, the dataset captures some of the linguistic variety that can be used to describe the same image.

Flickr8k is a good starting dataset as it is small in size and can be trained easily on low-end laptops/desktops using a CPU.

Dataset structure is as follows:

- drive/MyDrive/Machine Learning Project/
  - archive/
    - Images: contains the 8000+ images
  - Flickr8k\_Text/

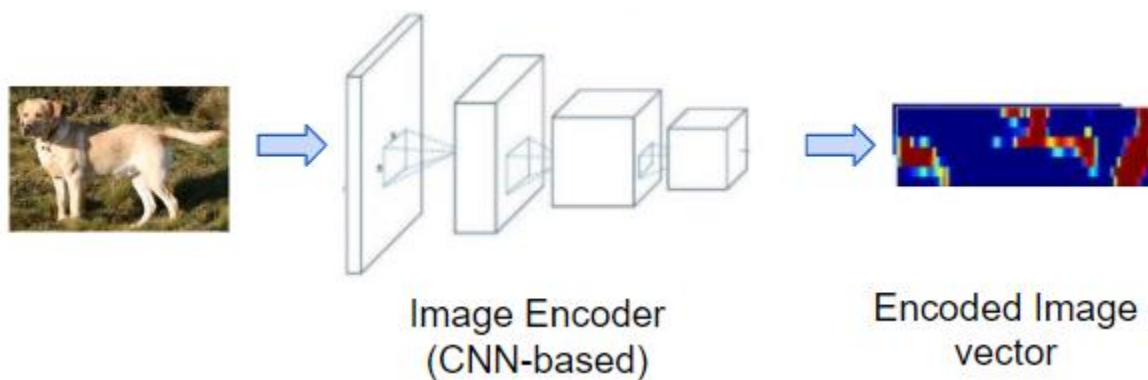
- Flickr8k.token.txt: contains the image id along with the captions
- Flickr8k.trainImages.txt: contains the training image id's
- Flickr8k.testImages.txt: contains the test image id's



# PROPOSED ARCHITECTURE

## Image Feature Encoder:

This takes the source photo as input and produces an encoded representation of it that captures its essential features.



This uses a CNN architecture and is usually done using transfer learning. We take a CNN model that was pre-trained for image classification and remove the final section which is the classifier. There are several such models like VGGNet, ResNet and Inception.

The backbone of this model consists of several CNN blocks which progressively extract various features from the photo, and generate a compact feature map that captures the most important elements in the picture.

Eg: It starts by extracting simple geometric shapes like curves and semi-circles in the initial layers, progresses to

higher-level structures such as noses, eyes, and hands, and eventually identifies elements such as faces and wheels.

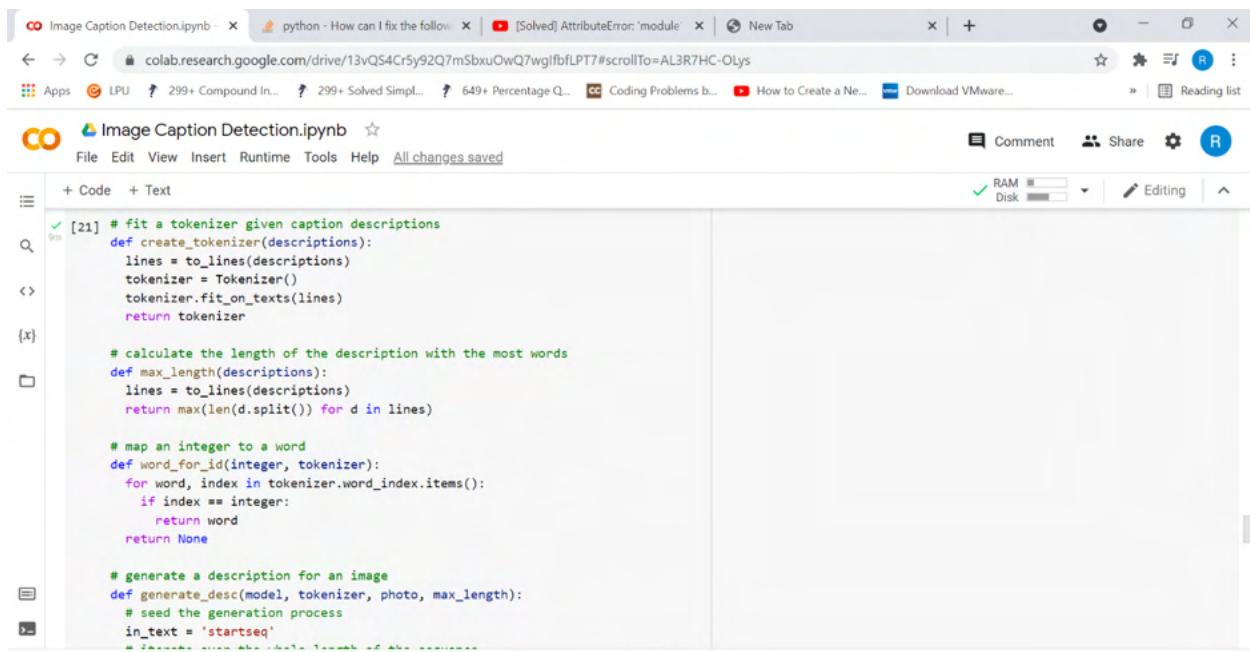
In an Image Classification model, this feature map is then fed to the last stage which is the Classifier that generates the final output prediction of the class of the primary object in the image.

When applying this model for Image Captioning, we are interested in the feature map representation of the image and do not need the classification prediction. So we keep the backbone and remove the classifier layers.

# RESULT AND EXPERIMENT ANALYSIS

BLEU is a metric for evaluating a generated sentence to a reference sentence. The score was developed for evaluating the predictions made by automatic machine translation systems. A perfect match results in a score of 1.0, whereas a perfect mismatch results in a score 0.0.

The actual and predicted descriptions are collected and evaluated using the corpus BLEU score that summarizes how close the generated text is to the expected text.



The screenshot shows a Jupyter Notebook titled "Image Caption Detection.ipynb" in a web browser. The code is as follows:

```
[21] # fit a tokenizer given caption descriptions
def create_tokenizer(descriptions):
    lines = to_lines(descriptions)
    tokenizer = Tokenizer()
    tokenizer.fit_on_texts(lines)
    return tokenizer

# calculate the length of the description with the most words
def max_length(descriptions):
    lines = to_lines(descriptions)
    return max(len(d.split()) for d in lines)

# map an integer to a word
def word_for_id(integer, tokenizer):
    for word, index in tokenizer.word_index.items():
        if index == integer:
            return word
    return None

# generate a description for an image
def generate_desc(model, tokenizer, photo, max_length):
    # seed the generation process
    in_text = 'startseq'
    # iterate over the data length of the sequence
```



Image Caption Detection.ipynb · python - How can I fix the follow... [Solved] AttributeError: 'module' x New Tab

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLys

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

### Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text RAM Disk Editing

```
[21] # load training dataset (6K)
filename = '/content/drive/MyDrive/Machine Learning Project/Flickr8k_text/Flickr_8k.trainImages.txt'
train = load_set(filename)
print('Dataset: %d' % len(train))
# descriptions
train_descriptions = load_clean_descriptions('descriptions.txt', train)
print('Descriptions: train=%d' % len(train_descriptions))
# prepare tokenizer
tokenizer = create_tokenizer(train_descriptions)
vocab_size = len(tokenizer.word_index) + 1
print('Vocabulary Size: %d' % vocab_size)
# determine the maximum sequence length
max_length = max_length(train_descriptions)
print('Description Length: %d' % max_length)

# prepare test set

# load test set
filename = '/content/drive/MyDrive/Machine Learning Project/Flickr8k_text/Flickr_8k.testImages.txt'
test = load_set(filename)
print('Dataset: %d' % len(test))
# descriptions
test_descriptions = load_clean_descriptions('descriptions.txt', test)
```

Image Caption Detection.ipynb python - How can I fix the follow... [Solved] AttributeError: 'module' x New Tab

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLys

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

### Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text RAM Disk Editing

```
[21] print('Dataset: %d' % len(test))
# descriptions
test_descriptions = load_clean_descriptions('descriptions.txt', test)
print('Descriptions: test=%d' % len(test_descriptions))
# photo features
test_features = load_photo_features('features.pkl', test)
print('Photos: test=%d' % len(test_features))
# load the model which has minimum loss, in this case it was model_18
filename = '/content/model_4.h5'
model = load_model(filename)
# evaluate model
evaluate_model(model, test_descriptions, test_features, tokenizer, max_length)
```

Dataset: 6000  
Descriptions: train=6000  
Vocabulary Size: 7579  
Description Length: 34  
Dataset: 1000  
Descriptions: test=1000  
Photos: test=1000  
WARNING:tensorflow:5 out of the last 8770 calls to <function Model.make\_predict\_function.<locals>.predict\_function at 0x7f45a233e8c0> triggered tf.func  
BLEU-1: 0.532145  
BLEU-2: 0.277866  
BLEU-3: 0.184814  
R1 Fil-A: 0.081184

9% completed at 0:03 PM



# SCREENSHOTS

```
[1] from google.colab import drive
drive.mount('/content/drive/')

Mounted at /content/drive/

from os import listdir
from pickle import dump
from keras.applications.vgg16 import VGG16
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from keras.applications.vgg16 import preprocess_input
from keras.models import Model

# extract features from each photo in the directory
def extract_features(directory):
    # load the model
    model = VGG16()
    # re-structure the model
    model = Model(inputs=model.inputs, outputs=model.layers[-2].output)
    # summarize
```

```
def extract_features(directory):
    # load the model
    model = VGG16()
    # re-structure the model
    model = Model(inputs=model.inputs, outputs=model.layers[-2].output)
    # summarize
    print(model.summary())
    # extract features from each photo
    features = dict()
    for name in listdir(directory):
        # load an image from file
        filename = directory + '/' + name
        image = load_img(filename, target_size=(224, 224))
        # convert the image pixels to a numpy array
        image = img_to_array(image)
        # reshape data for the model
        image = image.reshape((1, image.shape[0], image.shape[1], image.shape[2]))
        # prepare the image for the VGG model
        image = preprocess_input(image)
        # get features
        feature = model.predict(image, verbose=0)
        # get image id
        image_id = name.split('.')[0]
```

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=eMoXlaLcGkjq

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
# get image id
image_id = name.split('.')[0]
# store feature
features[image_id] = feature
print('%s' % name)
return features

# extract features from all images
directory = '/content/drive/MyDrive/Machine Learning Project/archive/Images'
features = extract_features(directory)
print('Extracted Features: %d' % len(features))
# save to file
dump(features, open('features.pkl', 'wb'))

>1077931201_1e0bb83105.jpg
>1075867198_27ca2e7efe.jpg
>108898978_7713be88fc.jpg
>1067180831_a59dc64344.jpg
>1077546505_a4f6c4daa9.jpg
>106490881_5a2dd9b7bd.jpg
>1067798824_f3cc97239b.jpg
>1082379191_ec1e53f996.jpg
>106514190_bae200f463.jpg
>1056359656_662cee0814.jpg
>1067675215_7336a694d6.jpg
```

Executing (8m 43s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_\_call\_\_() > \_call() > \_\_call\_\_() > \_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=eMoXlaLcGkjq

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

```
>1122944218_8eb3607403.jpg
>1119015538_e8e796281e.jpg
>1121416483_c7902d0d49.jpg
>1119463452_69d4eecd08.jpg
>1131155939_b4b457b05e.jpg
Extracted Features: 8116

[4] import string

# load doc into memory
def load_doc(filename):
    # open the file as read only
    file = open(filename, 'r')
    # read all text
    text = file.read()
    # close the file
    file.close()
    return text

# extract descriptions for images
def load_descriptions(doc):
    mapping = dict()
    # process lines
```

Executing (8m 43s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_\_call\_\_() > \_call() > \_\_call\_\_() > \_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=eMoXlaLcGkjq

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

```
[4] def load_descriptions(doc):
    mapping = dict()
    # process lines
    for line in doc.split('\n'):
        # split line by white space
        tokens = line.split()
        if len(tokens) < 2:
            continue
        # take the first token as the image id, the rest as the description
        image_id, image_desc = tokens[0], tokens[1:]
        # remove filename from image id
        image_id = image_id.split('.')[0]
        # convert description tokens back to string
        image_desc = ' '.join(image_desc)
        # create the list if needed
        if image_id not in mapping:
            mapping[image_id] = list()
        # store description
        mapping[image_id].append(image_desc)
    return mapping

def clean_descriptions(descriptions):
    # prepare translation table for removing punctuation
    table = str.maketrans('', '', string.punctuation)
```

Executing (9m 2s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_call\_\_() > \_call\_\_() > \_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=eMoXlaLcGkjq

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[4] def clean_descriptions(descriptions):
    # prepare translation table for removing punctuation
    table = str.maketrans('', '', string.punctuation)
    for key, desc_list in descriptions.items():
        for i in range(len(desc_list)):
            desc = desc_list[i]
            # tokenize
            desc = desc.split()
            # convert to lower case
            desc = [word.lower() for word in desc]
            # remove punctuation from each token
            desc = [w.translate(table) for w 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()]
            # store as string
            desc_list[i] = ' '.join(desc)

    # convert the loaded descriptions into a vocabulary of words
    def to_vocabulary(descriptions):
        # build a list of all description strings
        all_desc = set()
        for key, desc_list in descriptions.items():
            for desc in desc_list:
```

Executing (9m 2s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_call\_\_() > \_call\_\_() > \_call\_flat() > call() > quick\_execute()



Image Caption Detection.ipynb

colab.research.google.com/drive/13vQS4Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=eMoXIaLCGkjq

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk

Editing

[4] def to\_vocabulary(descriptions):  
# build a list of all description strings  
all\_desc = set()  
for key in descriptions.keys():  
[all\_desc.update(d.split()) for d in descriptions[key]]  
return all\_desc  
  
# save descriptions to file, one per line  
def save\_descriptions(descriptions, filename):  
lines = list()  
for key, desc\_list in descriptions.items():  
for desc in desc\_list:  
lines.append(key + ' ' + desc)  
data = '\n'.join(lines)  
file = open(filename, 'w')  
file.write(data)  
file.close()  
  
# filename = 'Flickr8k\_text/Flickr8k.token.txt'  
filename = '/content/drive/MyDrive/Machine Learning Project/Flickr8k\_text/Flickr8k.token.txt'  
# load descriptions  
doc = load\_doc(filename)  
# parse descriptions  
descriptions = load\_descriptions(doc)

Executing (9m 2s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_call\_\_() > \_call() > \_call\_\_() > \_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQS4Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=eMoXIaLCGkjq

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk

Editing

[4] # load descriptions  
doc = load\_doc(filename)  
# parse descriptions  
descriptions = load\_descriptions(doc)  
print('Loaded: %d ' % len(descriptions))  
# clean descriptions  
clean\_descriptions(descriptions)  
# summarize vocabulary  
vocabulary = to\_vocabulary(descriptions)  
print('Vocabulary Size: %d' % len(vocabulary))  
# save to file  
save\_descriptions(descriptions, 'descriptions.txt')  
  
Loaded: 8092  
Vocabulary Size: 8763  
  
[5] from pickle import load  
  
# load doc into memory  
def load\_doc(filename):  
# open the file as read only  
file = open(filename, 'r')  
# read all text  
text = file.read()

Executing (9m 2s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_call\_\_() > \_call() > \_call\_\_() > \_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=eMoXIaLCGkjq

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

RAM Disk

Editing

+ Code + Text

[5] def load\_doc(filename):  
# open the file as read only  
file = open(filename, 'r')  
# read all text  
text = file.read()  
# close the file  
file.close()  
return text  
  
# load a pre-defined list of photo identifiers  
def load\_set(filename):  
doc = load\_doc(filename)  
dataset = list()  
# process line by line  
for line in doc.split('\n'):  
# skip empty lines  
if len(line) < 1:  
continue  
# get the image identifier  
identifier = line.split('.')[0]  
dataset.append(identifier)  
return set(dataset)  
  
# load clean descriptions into memory

Executing (9m 2s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_call\_\_() > \_call() > \_call\_\_() > \_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=eMoXIaLCGkjq

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help Saving...

Comment Share

RAM Disk

Editing

+ Code + Text

[5] # load clean descriptions into memory  
def load\_clean\_descriptions(filename, dataset):  
# load document  
doc = load\_doc(filename)  
descriptions = dict()  
for line in doc.split('\n'):  
# split line by white space  
tokens = line.split()  
# split id from description  
image\_id, image\_desc = tokens[0], tokens[1:]  
# skip images not in the set  
if image\_id in dataset:  
# create list  
if image\_id not in descriptions:  
descriptions[image\_id] = list()  
# wrap description in tokens  
desc = 'startseq ' + ' '.join(image\_desc) + ' endseq'  
# store  
descriptions[image\_id].append(desc)  
return descriptions  
  
# load photo features  
def load\_photo\_features(filename, dataset):  
# load all features

Executing (9m 2s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_call\_\_() > \_call() > \_call\_\_() > \_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=f3ayAJs9LNsF

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
def load_photo_features(filename, dataset):
    # load all features
    all_features = load(open(filename, 'rb'))
    # filter features
    features = {k: all_features[k] for k in dataset}
    return features

# load training dataset (6K)
filename = '/content/drive/MyDrive/Machine Learning Project/Flickr8k_text/Flickr_8k.trainImages.txt'
train = load_set(filename)
print('Dataset: %d' % len(train))
# descriptions
train_descriptions = load_clean_descriptions('descriptions.txt', train)
print('Descriptions: train=%d' % len(train_descriptions))
# photo features
train_features = load_photo_features('features.pkl', train)
print('Photos: train=%d' % len(train_features))
train_descriptions
```

Dataset: 6000  
Descriptions: train=6000  
Photos: train=6000  
{'1000268201\_693b08c0e': ['startseq child in pink dress is climbing up set of stairs in an entry way endseq',  
 'startseq girl going into wooden building endseq',  
 ...]}

Executing (9m 41s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_\_call\_\_() > \_\_call\_\_() > \_\_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=f3ayAJs9LNsF

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

```
from numpy import array
import tensorflow
from pickle import load
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.utils import plot_model
from tensorflow.keras.models import Model
from keras.layers import Input
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Embedding
from keras.layers import Dropout
from keras.layers.merge import add
from keras.callbacks import ModelCheckpoint

# load doc into memory
def load_doc(filename):
    # open the file as read only
    file = open(filename, 'r')
    # read all text
    text = file.read()
    return text
```

Executing (9m 41s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_\_call\_\_() > \_\_call\_\_() > \_\_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=f3ayAJs9LNsF

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

RAM Disk

Editing

```
def load_doc(filename):
    # open the file as read only
    file = open(filename, 'r')
    # read all text
    text = file.read()
    # close the file
    file.close()
    return text

# load a pre-defined list of photo identifiers
def load_set(filename):
    doc = load_doc(filename)
    dataset = list()
    # process line by line
    for line in doc.split('\n'):
        # skip empty lines
        if len(line) < 1:
            continue
        # get the image identifier
        identifier = line.split('.')[0]
        dataset.append(identifier)
    return set(dataset)

# load clean descriptions into memory
```

Executing (9m 41s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_\_call\_\_() > \_call() > \_\_call\_\_() > \_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=f3ayAJs9LNsF

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
def load_clean_descriptions(filename, dataset):
    # load document
    doc = load_doc(filename)
    descriptions = dict()
    for line in doc.split('\n'):
        # split line by white space
        tokens = line.split()
        # split id from description
        image_id, image_desc = tokens[0], tokens[1:]
        # skip images not in the set
        if image_id not in dataset:
            # create list
            if image_id not in descriptions:
                descriptions[image_id] = list()
            # wrap description in tokens
            desc = 'startseq ' + ' '.join(image_desc) + ' endseq'
            # store
            descriptions[image_id].append(desc)
    return descriptions

# load photo features
def load_photo_features(filename, dataset):
    # load all features
    all_features = load_doc(filename, 'ph')
```

Executing (9m 41s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_\_call\_\_() > \_call() > \_\_call\_\_() > \_call\_flat() > call() > quick\_execute()



Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=f3ayAJs9LNsF

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
def load_photo_features(filename, dataset):
    # load all features
    all_features = load(open(filename, 'rb'))
    # filter features
    features = {k: all_features[k] for k in dataset}
    return features

# covert a dictionary of clean descriptions to a list of descriptions
def to_lines(descriptions):
    all_desc = list()
    for key in descriptions.keys():
        [all_desc.append(d) for d in descriptions[key]]
    return all_desc

# fit a tokenizer given caption descriptions
def create_tokenizer(descriptions):
    lines = to_lines(descriptions)
    tokenizer = Tokenizer()
    tokenizer.fit_on_texts(lines)
    return tokenizer

# get vocabulary size
tokenizer = create_tokenizer(train_descriptions)
```

Executing (9m 41s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_\_call\_\_() > \_\_call\_\_() > \_\_call\_\_() > \_\_call\_\_() > \_\_call\_\_() > \_\_call\_\_() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=f3ayAJs9LNsF

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
tokenizer = create_tokenizer(train_descriptions)
vocab_size = len(tokenizer.word_index) + 1
print('Vocabulary Size: %d' % vocab_size)

# calculate the length of the description with the most words
def max_length(descriptions):
    lines = to_lines(descriptions)
    return max(len(d.split()) for d in lines)

# create sequences of images, input sequences and output words for an image
def create_sequences(tokenizer, max_length, desc_list, photo):
    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=vocab_size)[0]
```

Executing (10m 26s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_\_call\_\_() > \_\_call\_\_() > \_\_call\_\_() > \_\_call\_\_() > \_\_call\_\_() > \_\_call\_\_() > quick\_execute()

colab.research.google.com/drive/13vQS4Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=f3ayAJs9LNsF

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

```
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
X1.append(photo)
X2.append(in_seq)
y.append(out_seq)
return array(X1), array(X2), array(y)

# define the captioning model
def define_model(vocab_size, max_length):
    # feature extractor model
    inputs1 = Input(shape=(4096,))
    fe1 = Dropout(0.5)(inputs1)
    fe2 = Dense(256, activation='relu')(fe1)
    # sequence model
    inputs2 = Input(shape=(max_length,))
    se1 = Embedding(vocab_size, 256, mask_zero=True)(inputs2)
    se2 = Dropout(0.5)(se1)
    se3 = LSTM(256)(se2)
    # decoder model
    decoder1 = add([fe2, se3])
    decoder2 = Dense(256, activation='relu')(decoder1)
    outputs = Dense(vocab_size, 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())
    return model
```

Executing (10m 33s) C > fit\_g... > error... > f > on\_train... > \_call\_ba... > \_call\_batc... > \_call\_batch... > on\_train... > \_batch\_upd... > sync\_to\_numpy... > map... > <lis... > \_to\_single\_numpy... > n... > \_nu... X

colab.research.google.com/drive/13vQS4Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=f3ayAJs9LNsF

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

```
outputs = Dense(vocab_size, 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())
return model
```

Vocabulary Size: 7579

```
#Below code is used to progressively load the batch of data
# data generator, intended to be used in a call to model.fit_generator()
def data_generator(descriptions, photos, tokenizer, max_length):
    # loop for ever over images
    while 1:
        for key, desc_list in descriptions.items():
            # retrieve the photo feature
            photo = photos[key][0]
            in_img, in_seq, out_word = create_sequences(tokenizer, max_length, desc_list, photo)
            yield [[in_img, in_seq], out_word]
```

```
[8] # load training dataset (6K)
```

Executing (10m 37s) Cell > fit\_generator() > error\_handler() > fit() > steps() > numpy() > read\_value() > \_read\_variable\_op() > read\_and\_set\_handle() > read\_variable\_op() X

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=f3ayAJs9LnSF

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

RAM

Disk

Editing

08

```
# load training dataset (6K)
filename = '/content/drive/MyDrive/Machine Learning Project/Flickr8k_text/Flickr_8k.trainImages.txt'
train = load_set(filename)
print('Dataset: %d' % len(train))
# descriptions
train_descriptions = load_clean_descriptions('descriptions.txt', train)
print('Descriptions: train=%d' % len(train_descriptions))
# photo features
train_features = load_photo_features('features.pkl', train)
print('Photos: train=%d' % len(train_features))
# prepare tokenizer
tokenizer = create_tokenizer(train_descriptions)
vocab_size = len(tokenizer.word_index) + 1
print('Vocabulary Size: %d' % vocab_size)
# determine the maximum sequence length
max_length = max_length(train_descriptions)
print('Description Length: %d' % max_length)
```

Dataset: 6000

Descriptions: train=6000

Photos: train=6000

Vocabulary Size: 7579

Description Length: 34

Executing (10m 41s) Cell > fit\_generator() > error\_handler() > fit() > error\_handler() > \_\_call\_\_() > \_call() > \_call\_\_() > \_call\_flat() > call() > quick\_execute()

Image Caption Detection.ipynb

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLys

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM

Disk

Editing

10

```
# train the model
model = define_model(vocab_size, max_length)
# train the model, run epochs manually and save after each epoch
epochs = 5
steps = len(train_descriptions)
for i in range(epochs):
    # create the data generator
    generator = data_generator(train_descriptions, train_features, tokenizer, max_length)
    # fit for one epoch
    model.fit_generator(generator, epochs=1, steps_per_epoch=steps, verbose=1)
    # save model
    model.save('model_' + str(i) + '.h5')
```

Model: "model\_1"

| Layer (type)          | Output Shape    | Param # | Connected to      |
|-----------------------|-----------------|---------|-------------------|
| input_3 (InputLayer)  | [(None, 34)]    | 0       | []                |
| input_2 (InputLayer)  | [(None, 4096)]  | 0       | []                |
| embedding (Embedding) | (None, 34, 256) | 1940224 | ['input_3[0][0]'] |

Executing (9m 32s) Cell > evaluate\_model() > generate\_desc() > error\_handler() > predict() > enumerate\_epochs() > \_\_iter\_\_() > \_\_init\_\_() > \_create\_iterator() > make\_iterator()

[9] Model: "model\_1"

| Layer (type)          | Output Shape    | Param # | Connected to                  |
|-----------------------|-----------------|---------|-------------------------------|
| input_3 (InputLayer)  | [(None, 34)]    | 0       | []                            |
| input_2 (InputLayer)  | [(None, 4096)]  | 0       | []                            |
| embedding (Embedding) | (None, 34, 256) | 1940224 | ['input_3[0][0]']             |
| dropout (Dropout)     | (None, 4096)    | 0       | ['input_2[0][0]']             |
| dropout_1 (Dropout)   | (None, 34, 256) | 0       | ['embedding[0][0]']           |
| dense (Dense)         | (None, 256)     | 1048832 | ['dropout[0][0]']             |
| lstm (LSTM)           | (None, 256)     | 525312  | ['dropout_1[0][0]']           |
| add (Add)             | (None, 256)     | 0       | ['dense[0][0]', 'lstm[0][0]'] |
| dense_1 (Dense)       | (None, 256)     | 65792   | ['add[0][0]']                 |
| dense_2 (Dense)       | (None, 7579)    | 1947803 | ['dense_1[0][0]']             |

Executing (9m 39s) Cell > evaluate\_model() > generate\_desc() > error\_handler() > predict() > error\_handler() > \_call\_\_() > \_call() > \_call\_\_() > \_call\_flat() > call() > quick\_execute()

[9]

|                 |              |         |                               |
|-----------------|--------------|---------|-------------------------------|
| add (Add)       | (None, 256)  | 0       | ['dense[0][0]', 'lstm[0][0]'] |
| dense_1 (Dense) | (None, 256)  | 65792   | ['add[0][0]']                 |
| dense_2 (Dense) | (None, 7579) | 1947803 | ['dense_1[0][0]']             |

=====  
 Total params: 5,527,963  
 Trainable params: 5,527,963  
 Non-trainable params: 0  
 =====

None

/usr/local/lib/python3.7/dist-packages/ipykernel\_launcher.py:10: UserWarning: 'Model.fit\_generator' is deprecated and will be removed in a future vers:  
 # Remove the CWD from sys.path while we load stuff.

6000/6000 [=====] - 1086s 180ms/step - loss: 4.6632

/usr/local/lib/python3.7/dist-packages/keras/engine/functional.py:1410: CustomMaskWarning: Custom mask layers require a config and must override get\_cc  
 layer\_config = serialize\_layer\_fn(layer)

6000/6000 [=====] - 1066s 178ms/step - loss: 3.8987

6000/6000 [=====] - 1052s 175ms/step - loss: 3.6552

6000/6000 [=====] - 1052s 175ms/step - loss: 3.4977

6000/6000 [=====] - 1056s 176ms/step - loss: 3.3914

4

Executing (9m 47s) Cell > evaluate\_model() > generate\_desc() > error\_handler() > predict() > error\_handler() > \_call\_\_() > \_call() > \_call\_\_() > \_call\_flat() > call() > quick\_execute()



Image Caption Detection.ipynb × python - How can I fix the follow... [Solved] AttributeError: 'module' × New Tab

colab.research.google.com/drive/13vQS4Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLys

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
#Entire code with all the functions, for reference purpose

from numpy import argmax
from pickle import load
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import load_model
from nltk.translate.bleu_score import corpus_bleu

# load doc into memory
def load_doc(filename):
    # open the file as read only
    file = open(filename, 'r')
    # read all text
    text = file.read()
    # close the file
    file.close()
    return text

# load a pre-defined list of photo identifiers
def load_set(filename):
    doc = load_doc(filename)
    dataset = list()
```

Image Caption Detection.ipynb × python - How can I fix the follow... [Solved] AttributeError: 'module' × New Tab

colab.research.google.com/drive/13vQS4Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLys

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
[21] dataset = list()
# process line by line
for line in doc.split('\n'):
    # skip empty lines
    if len(line) < 1:
        continue
    # get the image identifier
    identifier = line.split('.')[0]
    dataset.append(identifier)
return set(dataset)

# load clean descriptions into memory
def load_clean_descriptions(filename, dataset):
    # load document
    doc = load_doc(filename)
    descriptions = dict()
    for line in doc.split('\n'):
        # split line by white space
        tokens = line.split()
        # split id from description
        image_id, image_desc = tokens[0], tokens[1:]
        # skip images not in the set
        if image_id not in dataset:
            # create list
```

Image Caption Detection.ipynb · python - How can I fix the follow... [Solved] AttributeError: 'module' New Tab

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLys

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

✓ [21] # create list

```
if image_id not in descriptions:
    descriptions[image_id] = list()
# wrap description in tokens
desc = 'startseq ' + ' '.join(image_desc) + ' endseq'
# store
descriptions[image_id].append(desc)
return descriptions

# load photo features
def load_photo_features(filename, dataset):
    # load all features
    all_features = load(open(filename, 'rb'))
    # filter features
    features = {k: all_features[k] for k in dataset}
    return features

# covert a dictionary of clean descriptions to a list of descriptions
def to_lines(descriptions):
    all_desc = list()
    for key in descriptions.keys():
        [all_desc.append(d) for d in descriptions[key]]
    return all_desc
```

Image Caption Detection.ipynb · python - How can I fix the follow... [Solved] AttributeError: 'module' New Tab

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLys

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

✓ [21] # fit a tokenizer given caption descriptions

```
def create_tokenizer(descriptions):
    lines = to_lines(descriptions)
    tokenizer = Tokenizer()
    tokenizer.fit_on_texts(lines)
    return tokenizer

# calculate the length of the description with the most words
def max_length(descriptions):
    lines = to_lines(descriptions)
    return max(len(d.split()) for d in lines)

# map an integer to a word
def word_for_id(integer, tokenizer):
    for word, index in tokenizer.word_index.items():
        if index == integer:
            return word
    return None

# generate a description for an image
def generate_desc(model, tokenizer, photo, max_length):
    # seed the generation process
    in_text = 'startseq'
```

Image Caption Detection.ipynb · python - How can I fix the follow... [Solved] AttributeError: 'module' x New Tab

colab.research.google.com/drive/13vQS4Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLys

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Editing

```
[21] def generate_desc(model, tokenizer, photo, max_length):
# seed the generation process
in_text = 'startseq'
# iterate over the whole length of the sequence
for i in range(max_length):
# integer encode input sequence
sequence = tokenizer.texts_to_sequences([in_text])[0]
# pad input
sequence = pad_sequences([sequence], maxlen=max_length)
# predict next word
yhat = model.predict([photo, sequence], verbose=0)
# convert probability to integer
yhat = argmax(yhat)
# map integer to word
word = word_for_id(yhat, tokenizer)
# stop if we cannot map the word
if word is None:
break
# append as input for generating the next word
in_text += ' ' + word
# stop if we predict the end of the sequence
if word == 'endseq':
break
return in_text
```

Image Caption Detection.ipynb · python - How can I fix the follow... [Solved] AttributeError: 'module' x New Tab

colab.research.google.com/drive/13vQS4Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLys

Apps LPU 299+ Compound In... 299+ Solved Simpl... 649+ Percentage Q... Coding Problems b... How to Create a Ne... Download VMware... Reading list

Image Caption Detection.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Editing

```
[21] break
return in_text

# evaluate the skill of the model
def evaluate_model(model, descriptions, photos, tokenizer, max_length):
actual, predicted = list(), list()
# step over the whole set
for key, desc_list in descriptions.items():
# generate description
yhat = generate_desc(model, tokenizer, photos[key], max_length)
# store actual and predicted
references = [d.split() for d in desc_list]
actual.append(references)
predicted.append(yhat.split())
# calculate 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)))
print('BLEU-3: %f' % corpus_bleu(actual, predicted, weights=(0.3, 0.3, 0.3, 0)))
print('BLEU-4: %f' % corpus_bleu(actual, predicted, weights=(0.25, 0.25, 0.25, 0.25)))

# prepare training set

# load training dataset (6K)
```

Image Caption Detection.ipynbpython - How can I fix the follow[Solved] AttributeError: 'module'New Tabcolab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLysAppsLPU299+ Compound In...299+ Solved Simpl...649+ Percentage Q...Coding Problems b...How to Create a Ne...Download VMware...Reading list

Image Caption Detection.ipynbCommentShareAll changes savedFile Edit View Insert Runtime Tools Help

+ Code + TextRAMDiskEditing

[21] # load training dataset (6K)  
filename = '/content/drive/MyDrive/Machine Learning Project/Flickr8k\_text/Flickr\_8k.trainImages.txt'  
train = load\_set(filename)  
print('Dataset: %d' % len(train))  
# descriptions  
train\_descriptions = load\_clean\_descriptions('descriptions.txt', train)  
print('Descriptions: train=%d' % len(train\_descriptions))  
# prepare tokenizer  
tokenizer = create\_tokenizer(train\_descriptions)  
vocab\_size = len(tokenizer.word\_index) + 1  
print('Vocabulary Size: %d' % vocab\_size)  
# determine the maximum sequence length  
max\_length = max\_length(train\_descriptions)  
print('Description Length: %d' % max\_length)  
  
# prepare test set  
  
# load test set  
filename = '/content/drive/MyDrive/Machine Learning Project/Flickr8k\_text/Flickr\_8k.testImages.txt'  
test = load\_set(filename)  
print('Dataset: %d' % len(test))  
# descriptions  
test\_descriptions = load\_clean\_descriptions('descriptions.txt', test)

Image Caption Detection.ipynbpython - How can I fix the follow[Solved] AttributeError: 'module'New Tabcolab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLysAppsLPU299+ Compound In...299+ Solved Simpl...649+ Percentage Q...Coding Problems b...How to Create a Ne...Download VMware...Reading list

Image Caption Detection.ipynbCommentShareAll changes savedFile Edit View Insert Runtime Tools Help

+ Code + TextRAMDiskEditing

[21] print('Dataset: %d' % len(test))  
# descriptions  
test\_descriptions = load\_clean\_descriptions('descriptions.txt', test)  
print('Descriptions: test=%d' % len(test\_descriptions))  
# photo features  
test\_features = load\_photo\_features('features.pkl', test)  
print('Photos: test=%d' % len(test\_features))  
# load the model which has minimum loss, in this case it was model\_18  
filename = '/content/model\_4.h5'  
model = load\_model(filename)  
# evaluate model  
evaluate\_model(model, test\_descriptions, test\_features, tokenizer, max\_length)  
  
Dataset: 6000  
Descriptions: train=6000  
Vocabulary Size: 7579  
Description Length: 34  
Dataset: 1000  
Descriptions: test=1000  
Photos: test=1000  
WARNING:tensorflow:5 out of the last 8770 calls to <function Model.make\_predict\_function.<locals>.predict\_function at 0x7f45a233e8c0> triggered tf.func  
BLEU-1: 0.532145  
BLEU-2: 0.277866  
BLEU-3: 0.184814  
RI F1=0.4081184



colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=AL3R7HC-OLys

Image Caption Detection.ipynb

File Edit View Insert Runtime Tools Help

+ Code + Text

```
from pickle import load
from numpy import argmax
from keras.preprocessing.sequence import pad_sequences
from keras.applications.vgg16 import VGG16
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from keras.applications.vgg16 import preprocess_input
from keras.models import Model
from keras.models import load_model

# extract features from each photo in the directory
def extract_features(filename):
    # load the model
    model = VGG16()
    # re-structure the model
    model.layers.pop()
    model = Model(inputs=model.inputs, outputs=model.layers[-1].output)
    # load the photo
    image = load_img(filename, target_size=(224, 224))
    # convert the image pixels to a numpy array
    image = img_to_array(image)
    # reshape data for the model
    image = image.reshape((1, image.shape[0], image.shape[1], image.shape[2]))

# load the photo
image = load_img(filename, target_size=(224, 224))
# convert the image pixels to a numpy array
image = img_to_array(image)
# reshape data for the model
image = image.reshape((1, image.shape[0], image.shape[1], image.shape[2]))
# prepare the image for the VGG model
image = preprocess_input(image)
# get features
feature = model.predict(image, verbose=0)
return feature

# load the tokenizer
tokenizer = load(open('/content/features.pkl', 'rb'))
# pre-define the max sequence length (from training)
max_length = 34
# load the model
model = load_model('/content/model_4.h5')
# load and prepare the photograph
photo = extract_features('/content/drive/MyDrive/Machine Learning Project/sample images/sample image1.jpg')
# generate description
description = generate_desc(model, tokenizer, photo, max_length)
print(description)
```

Image Caption Detection.ipynb ×python - How can I fix the follow... ×[Solved] AttributeError: 'module' ×New Tab ×+ ×- ×

colab.research.google.com/drive/13vQ54Cr5y92Q7mSbxuOwQ7wglfbfLPT7#scrollTo=-R86fpeiOSgL

AppsLPU299+ Compound In...299+ Solved Simpl...649+ Percentage Q...Coding Problems b...How to Create a Ne...Download VMware...Reading list

Image Caption Detection.ipynb ☆

File Edit View Insert Runtime Tools Help

+ Code + Text

RAMDiskEditing

startseq man rides bike on dirty path endseq

#Remove startseq and endseq  
query = description  
stopwords = ['startseq', 'endseq']  
querywords = query.split()  
  
resultwords = [word for word in querywords if word.lower() not in stopwords]  
result = ' '.join(resultwords)  
  
print(result)

man rides bike on dirty path

# CONCLUSION AND FUTURE SCOPE AND REFERENCES

We have implemented a CNN-LSTM model for building an Image Caption Generator. A CNN-LSTM architecture has wide-ranging applications which include use cases in Computer Vision and Natural Processing domains.

Image Captioning has various applications such as recommendations in editing applications, usage in virtual assistants for image indexing, for visually impaired persons, for social media, and several other natural language processing applications. Recently, deep learning methods have achieved state-of-the-art results on examples of this problem. It has been demonstrated that deep learning models are able to achieve optimum results in the field of caption generation problems.

References:

1. [https://www.ripublication.com/ijaer18/ijaerv13n9\\_102.pdf](https://www.ripublication.com/ijaer18/ijaerv13n9_102.pdf)
2. <https://cs.stanford.edu/people/karpathy/cvpr2015.pdf>
3. <https://arxiv.org/pdf/1411.4555.pdf>

-----THE END-----