ImageSpeak: Generating Captions from Pixels

A PROJECT REPORT

Submitted by

Shivani 21BCS6285
Ashmandeep Kaur 21BCS6284
Tarushi Sandeep Gupta 21BCS6280

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE WITH SPECIALIZATION IN
ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Chandigarh University
NOV 2023

BONAFIDE CERTIFICATE

Certified that this project report **“ ImageSpeak: Generating Captions from Pixels ”** is the bonafide work of “ **Shivani, Ashmandeep Kaur, and Tarushi Sandeep Gupta ”** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Mr. Aman kaushik	Mr.Nirmalya Basu
HEAD OF THE DEPARTMENT	**SUPERVISOR**
AIT CSE	AIT CSE

Submitted for the project viva-voce examination held on__________

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

TABLE OF CONTENTS

Abstract………………………………………………………………………………..5

Graphical Abstract………………………………………………………..……………6

Abbreviations……………………………………………………………………………...8

Chapter 1

Introduction…………………………………………………………….…...........9

1.1. Problem Definition …………………………….………………………….…………10

1.2. Problem Overview……………………………...…………………………….11

1.3. Hardware Specifications ……………………………………………………..12

1.4.SoftwareSpecifications.…………………………………………….……………12

1.5.Dataset Used ……………………….....……………………………….…12

Chapter 2

Literature review……………………….………………………………….…14

2.1. Literature Survey……………..……………………………………………14

2.2. Existing System…………………………………………………………….23

2.3. Proposed solutions ……..…………………………………………………..24

Chapter 3

Design Flow/Process……………………...………….………………………27

3.1. Experimental Setup……………………………..….…………………………28

3.2. Challenges…………………………………………………………………….31

3.3. Objectives ………………...………………………………………..………32

Chapter 4

Results analysis and validation……………………………………………………..34

4.1. Model Validation……………………..….…………………………………..36

4.2. Result……………………………………………………………………………...39

Chapter 5

Conclusion and future work……………………………………………………...41

5.1. Conclusion…………………………../……………………………………………41

5.2. Future work………………………………………………………………………...42

References………………….…………………………………………………………….44

User Manual……………………………………………………………………………….48

ABSTRACT

In the ImageSpeak project, we introduce a ground breaking application of artificial intelligence that combines computer vision and natural language processing. Our goal is to empower machines to understand visuals and generate relevant captions with ease. Our innovative solution effectively overcomes the challenge of translating images into descriptive text and has wide-reaching potential across various fields. Our method integrates convolution neural networks (CNNs) for image analysis and recurrent neural networks (RNNs) for language generation, resulting in a robust system capable of perceiving and accurately describing complex visual scenes. We have extensively trained our model on diverse and extensive datasets consisting of images paired with real human-generated captions.

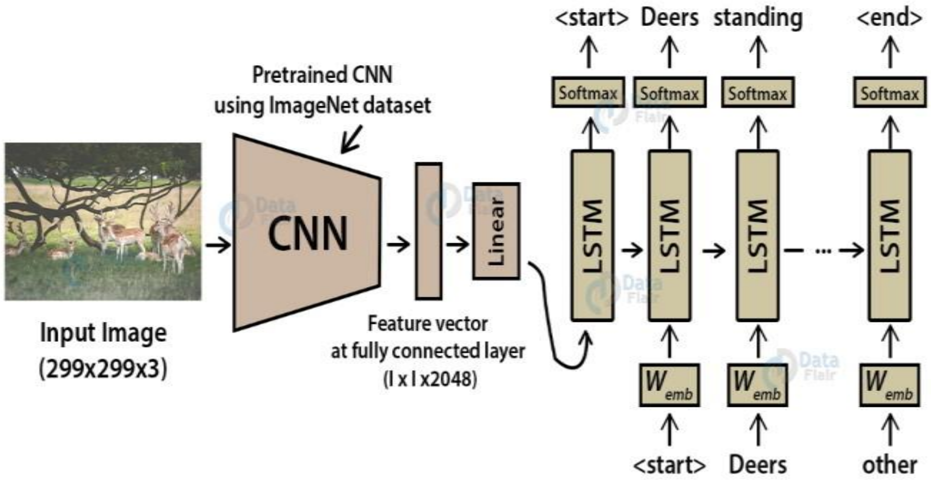
The project develops a coherent system that can recognize and describe complex visual situations by utilizing the capabilities of recurrent neural networks (RNNs) for language synthesis and convolutional neural networks (CNNs) for image processing. The project's main goal is to train the model using sizable and varied datasets that include pictures and annotations written by humans. By means of comprehensive training and optimization, the model acquires the ability to identify visual characteristics and correlate them with suitable spoken phrases.

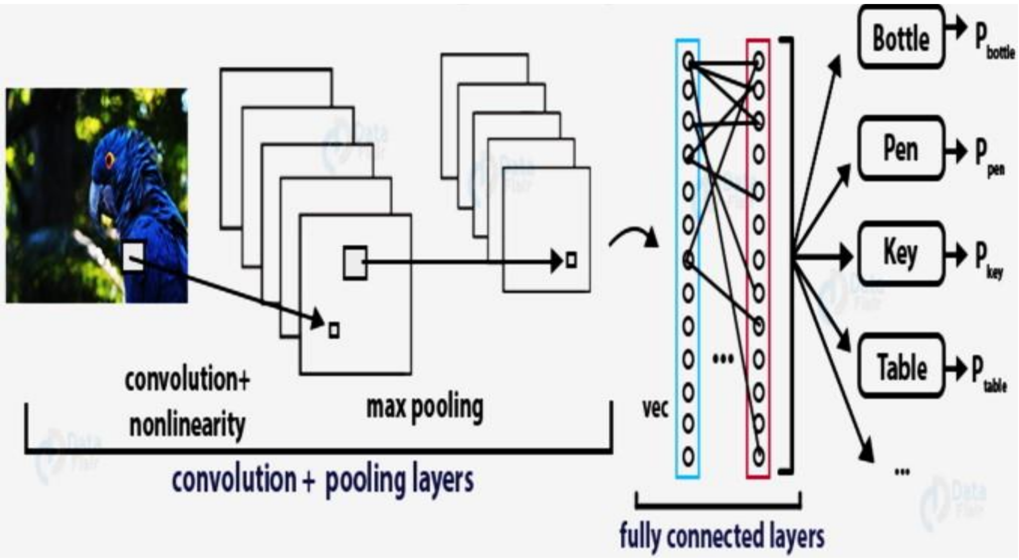
The produced captions attempt to portray the emotions, relationships within the setting, and the spirit of the image, going beyond simple factual explanations. Applications like as helping the blind and visually challenged, boosting search engine performance, and increasing social media participation are made possible by this contextual awareness.

To sum up, the ImageSpeak project is a perfect example of how computer vision and natural language processing work together, highlighting the revolutionary ways that AI-driven technologies are improving the field of human-computer interaction. The project's results provide a foundation for future developments, promoting the creation of multimodal AI systems that can understand and produce a variety of digital information.

GRAPHICAL ABSTRACT

1. Input Image: An array of pixels representing diverse visual content.
2. Preprocessing: Image undergoes preprocessing steps like resizing, normalization, and feature extraction using CNN layers.
3. Convolutional Neural Network (CNN): Layers of convolutional and pooling operations extract hierarchical features from the image.
4. Feature Representation: CNN encodes the image into a high-dimensional feature vector, capturing spatial and semantic information.
5. Sequence Generation: The feature vector serves as input to a sequence generation model like LSTM (Long Short-Term Memory) or Transformer, generating descriptive captions.
6. Language Model Training: Model learns to associate image features with textual descriptions using datasets with image-caption pairs.
7. Caption Prediction: Generated captions undergo evaluation based on relevance, coherence, and grammar.
8. Output: Accurate and contextually relevant captions describing the content of the input image.
9. Applications: Enables automated image description for visually impaired individuals, enhances image search engines, and aids in content retrieval.
10. Future Directions: Improving model robustness, exploring multi-modal architectures for better understanding visual context.
11. Conclusion: CNN-based image captioning systems bridge the gap between visual content and natural language understanding, fostering applications across various domains.


Pretrained CNN
using ImageNet dataset
CNN
Linear
Input Image
(299x299x3)
Feature vector
at fully connected layer
(l x l x2048)
<start>
Deers
standing
<end>
Softmax
Softmax
Softmax
Softmax
LSTM
LSTM
LSTM
LSTM
W_{emb}
W_{emb}
W_{emb}
<start>
Deers
other


convolution+
nonlinearity
max pooling
convolution + pooling layers
vec
fully connected layers
Bottle
P_{bottle}
Pen
P_{pen}
Key
P_{key}
Table
P_{table}
...

ABBREVIATIONS

- CNN - Convolutional Neural Network
- RNN - Recurrent Neural Network
- LSTM - Long Short-Term Memory
- NLP - Natural Language Processing
- BERT - Bidirectional Encoder Representations from Transformers
- GPU - Graphics Processing Unit
- GPT - Generative Pre-trained Transformer
- IDE - Integrated Development Environment
- ResNet - Residual Network
- VGG - Visual Geometry Group
- METEOR - Metric for Evaluation of Translation with Explicit Ordering of Hypotheses
- BLEU - Bilingual Evaluation Understudy
- ROUGE - Recall-Oriented Understudy for Gisting Evaluation

CHAPTER 1
INTRODUCTION

The emergence of Imagespeak within the realm of artificial intelligence represents a remarkable stride in bridging the gap between visual content and human language. It's a concept that revolves around the remarkable capacity of machines to not just perceive visual data but also articulate it into easily comprehensible human language. At its core, Imagespeak embodies the fusion of sophisticated algorithms and deep learning methodologies, offering machines the ability to comprehend and communicate the content depicted in images.

Convolutional neural networks (CNNs), a potent subclass of deep learning models created especially to evaluate visual input, are essential to Imagespeak's operation. These neural networks, which effectively imitate some parts of human visual perception, are highly skilled at identifying complex patterns, forms, textures, and characteristics inside pictures. These CNNs are trained to analyze and interpret visual information by means of intensive training on large datasets where photos are carefully matched with descriptive phrases or captions.

The AI algorithms process a vast number of image-caption pairings during the training phase. They use picture analysis to identify objects, deduce context, and comprehend the connections between different visual data sources. In essence, this procedure trains the AI to link particular visual clues to related language descriptions.

Through iterative learning, the AI comprehends a diverse range of visual elements and their representations in human language. Once the training phase concludes, the model is equipped with a profound understanding of these visual-to-language mappings. When presented with a new image, the model can draw upon its learned knowledge to interpret the visual content and generate descriptive captions that are not only accurate but also contextually meaningful. This ability of the AI to perceive an image, extract its visual features, and articulate them into coherent and relevant textual descriptions represents a pivotal milestone in the evolution of AI capabilities.

Essentially, Imagespeak uses CNNs' extraordinary strength and the combination of large datasets to give machines the capacity to not only "see" images but also to interpret their contents in a language that is understandable to humans. This invention has exciting potential applications in a number of fields, including content enrichment and accessibility aids as well as the transformation of how we interact with and understand visual material in our increasingly digital society.

1.1 Problem Definition

Creating an image caption generator is a fascinating blend of computer vision and natural language processing. Its core task is twofold: understanding the content of an image and translating that understanding into coherent, descriptive captions. The model undergoes a complex process involving visual recognition and language generation.

To start, the model needs to comprehend the image's elements. This includes recognizing objects, people, animals, or any significant features within the image using techniques like convolutional neural networks (CNNs). Additionally, understanding the broader scene—identifying locations, backgrounds, and the relationships between objects—is essential. Furthermore, recognizing activities or actions occurring in the image, such as walking, playing, or eating, contributes to a more comprehensive understanding. Once the image is understood, the model transitions to natural language generation. It structures this comprehension into coherent sentences that accurately describe the scene. Generating grammatically correct and fluent sentences becomes crucial at this stage. The captions should not only describe what is visible in the image but also convey the relationships between objects and their significance within the context.

Training such a model requires a robust dataset with images paired with human-generated captions. These pairs serve as inputs and target outputs for the model during training. Architectures combining image processing techniques, such as CNNs, with language generation models, like recurrent neural networks (RNNs) or transformers, are employed to create an end-to-end system. Evaluating the quality of generated captions involves various metrics. BLEU and METEOR metrics assess similarity to reference captions, fluency, and accuracy, while human evaluation measures overall coherence and relevance. Handling ambiguity within images and ensuring adaptability to diverse image types and scenarios are ongoing challenges for such models.

The goal is to bridge the gap between visual information and textual description, making visual content more accessible and comprehensible. Models like ImageSpeak have made significant strides in this field. They utilize advanced architectures like vision transformers and multimodal learning to enhance image understanding and caption generation, resulting in more accurate and contextually relevant captions. This progress benefits various applications, including image indexing, accessibility tools, and aiding visually impaired individuals.

1.2 Problem Overview

The ImageSpeak project endeavors to tackle a critical challenge empowering machines to comprehend and articulate visual content. While humans effortlessly interpret images, translating this innate ability into AI systems remains a formidable hurdle. Existing methods often fall short in capturing the intricate details of visual scenes, resulting in superficial and often inaccurate descriptions. The crux of the issue lies in the disparity between the understanding of images and the expression of that understanding in language. Computers struggle to extract contextual nuances, emotions, and the intricate relationships embedded within images, limiting their capacity to generate meaningful captions.

This limitation poses barriers across various domains, including accessibility tools, e-commerce, and content enhancement, hindering progress in these fields. An additional concern pertains to biases present in training data, which can permeate into the generated captions, perpetuating stereotypes and disseminating misinformation. Therefore, ethical considerations hold immense importance in the development of responsible AI systems, necessitating careful handling and mitigation of biases ingrained in the training process.

The project's overarching goal encompasses the creation of ImageSpeak: Generating Captions from Pixels—a system that harnesses neural networks by amalgamating computer vision for intricate image analysis and natural language processing for caption generation. Through training on diverse datasets, this system aims to acquire the capability to discern visual features and translate them into coherent, contextually relevant textual descriptions. Beyond accuracy, the emphasis lies on ensuring ethical and unbiased caption generation. Striving for fairness and contextual understanding becomes pivotal in crafting captions that not only describe images accurately but also do so responsibly, avoiding perpetuation of stereotypes or dissemination of false information.

In essence, the ImageSpeak project strives to bridge the chasm between visuals and language, poised to revolutionize the way AI interacts with and interprets visual content. By achieving this, the project aims to contribute significantly to a more comprehensive and inclusive digital environment, fostering advancements across multiple domains while upholding ethical standards in AI development and deployment.

1.3 Hardware Specification

High performance CPUs GPUs : CPUs like Intel Core i9 or AMD Ryzen 9, and GPUs like NVIDIA GeForce RTX 30 series or AMD Radeon RX 6000 series .

Memory(RAM): A minimum of 16 GB to 32 GB of RAM is recommended to handle large data and complex images.

Reliable Internet Connection A stable, high-speed Internet connection is required.

1.4 Software Specification

Programming Language: There are many different programming languages Which can be used for ML. Python is the most commonly used language for ML.It had a rich ecosystem of libraries and data mining tools,Images and ML. So it is a good choice for developing an image caption generator

Integrated Development Environment (IDEs): used are Visual Studio Code, Google Collab and Jupyter Notebook.

1.5 Dataset Used

The foundation of this research rests upon the utilization of the "Flickr 8k" dataset, a widely acknowledged benchmark within the field of image captioning. Comprising 8,000 images sourced from the Flickr platform, this dataset stands as a cornerstone for evaluating and training models dedicated to generating descriptive captions for images. What sets this dataset apart is the richness it offers in diversity and granularity through the provision of five distinct captions for each of the 8,000 images.

Each image within the "Flickr 8k" dataset is accompanied by a set of five distinct and descriptive captions. This multiplicity of captions serves as a crucial asset, fostering the training and evaluation of models by providing varied and detailed descriptions. This diversity enables models to encapsulate a spectrum of contexts, encompassing different scenes, objects, activities, and compositions prevalent across the images. This variance empowers models to comprehend and generate captions that are nuanced and versatile, reflecting the multifaceted nature of visual content present in real-world scenarios.

The compilation of images within the "Flickr 8k" dataset encapsulates a broad spectrum of subjects and scenes. This inclusivity is instrumental in training models to understand and generate captions across a wide array of scenarios. The diversity within the dataset ensures that models are exposed to various visual elements, from everyday scenes to diverse compositions, enabling them to grasp the intricacies of image content representation. This broad exposure aids in enhancing the models' ability to comprehend and articulate detailed, contextually relevant captions.

In essence, the "Flickr 8k" dataset's richness lies not just in its expansive collection of images but also in the multitude of captions accompanying each image. This wealth of descriptive information empowers models in their learning process, allowing them to capture the nuances of diverse visual contexts, ultimately leading to the development of more accurate, context-aware, and comprehensive image captioning models.

CHAPTER 2
LITERATURE SURVEY

1. “Show and Tell: A Neural Image Caption Generator” by Oriol Vinyals et al
   Tools Used : Tensorflow, Python, Numpy, Pandas
   Technique used : CNN, RNN, Attention mechanism

Show and Tell," an influential work by Oriol Vinyals and colleagues, stands as a pioneering milestone in the realm of automated image captioning. At its core, this groundbreaking system ingeniously amalgamates two fundamental neural network architectures: Convolutional Neural Networks (CNNs) and Long Short-Term Memory networks (LSTMs). The system's architectural brilliance lies in its strategic utilization of CNNs for processing visual information and LSTMs for generating descriptive captions.

CNNs, renowned for their prowess in visual recognition, serve as the initial stage of the process. These networks meticulously analyze images, extracting intricate visual features layer by layer. This hierarchical feature extraction culminates in encoding the image's essence into a compact, fixed-length vector.

Subsequently, the torch is passed to the LSTM networks, which specialize in understanding sequential data and generating coherent, contextually rich text. Armed with the encoded visual representation, the LSTM network seamlessly translates these visual cues into comprehensive and contextually relevant captions. Through a trained understanding of sequential patterns and language structures derived from extensive image-caption datasets, the LSTM network becomes adept at articulating descriptions that accurately encapsulate the content and context of the input image.

What sets "Show and Tell" apart is its transformative fusion of visual and linguistic understanding within a unified deep learning framework. This amalgamation of CNNs, LSTMs, and attention mechanisms marks a pivotal leap in the domain of automated image captioning. The technology not only bridges the gap between image understanding and natural language generation but also revolutionizes the very landscape of how machines comprehend and communicate visual content.

The deployment of attention mechanisms further enriches this system, allowing it to dynamically focus on salient regions within an image while generating captions. This attention to relevant visual cues enhances the quality and relevance of the generated descriptions, contributing to more precise and nuanced captioning.

"Show and Tell" heralds a new era in automated image captioning, opening vistas of possibilities across numerous fields. From aiding visually impaired individuals by providing detailed verbal descriptions of images to enhancing search engine capabilities and facilitating content retrieval, its impact resonates widely across various applications. Its underlying technology forms the bedrock for advancing multimodal learning and fostering deeper synergies between visual understanding and natural language processing, thereby reshaping the landscape of automated image captioning.

2. “Neural Image Caption Generation with Visual Attention” by Kelvin Xu et al
   Tools Used : Deep learning, Frameworks, Cuda and cuDNN. LaTeX
   Technique used : CNN, RNN, Softmax Activation, BLEU

The paper on "Neural Image Caption Generation with Visual Attention" stands as a pivotal advancement in the domain of automated image captioning, introducing an innovative architecture that integrates neural networks with attention mechanisms. At its core, this revolutionary approach seamlessly combines the prowess of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) with the incorporation of visual attention mechanisms.

The CNN serves as the image encoder, adeptly capturing and encoding intricate visual features from the input image. This encoding process distills the image's essence into a condensed representation, setting the stage for the subsequent generation of descriptive captions.

The spotlight, however, falls on the integration of recurrent neural networks equipped with attention mechanisms. Unlike conventional systems, this architecture doesn't merely generate captions based on the entire encoded image representation. Instead, it dynamically focuses on specific regions of interest within the image during the caption generation process.

This innovation is made possible through the incorporation of attention mechanisms, which mimic human visual attention by selectively concentrating on relevant parts of the image while generating captions. By learning to attend to specific regions within the encoded image representation, the model hones in on crucial visual elements that significantly contribute to the descriptive richness and accuracy of the generated captions.

This selective attention mechanism revolutionizes image captioning in several profound ways. First and foremost, it enhances the quality of generated captions by incorporating pertinent visual information into the textual descriptions. This nuanced integration of visual cues leads to more accurate and detailed captions that intricately capture the essence of the image content.

Moreover, the model's ability to dynamically attend to relevant regions within the image during the captioning process contributes to a more contextually coherent and focused description. This selective concentration on image elements ensures that the generated captions are not only descriptive but also closely aligned with the semantic content and spatial arrangement of the visual scene.

The incorporation of visual attention mechanisms marks a paradigm shift in image captioning, transcending traditional approaches by enabling the model to intelligently focus on specific image elements while generating accurate, contextually relevant, and detailed descriptions. This pioneering architecture paves the way for a deeper understanding of the symbiotic relationship between visual perception and linguistic expression, setting new standards for the fusion of visual and textual modalities in AI-driven image understanding and communication.

3. “Knowing When to Look : Adaptive Attention via A Visual Sentinel for Image Captioning” by Jiasen Lu et al
   Technique used : Reinforcement Learning, attention mechanism

"Knowing When to Look: Adaptive Attention via A Visual Sentinel for Image Captioning" presents a groundbreaking approach in the realm of image captioning, leveraging an adaptive attention mechanism that introduces the concept of a "visual sentinel" to guide the model's focus during the caption generation process.

This innovative methodology harmoniously combines the capabilities of Convolutional Neural Networks (CNNs) for extracting rich visual features from images and Recurrent Neural Networks (RNNs) for language modeling. However, the defining feature lies in the integration of attention mechanisms guided by a visual sentinel.

The visual sentinel acts as a guiding force for the model, dynamically determining which image regions deserve attention and which should be disregarded, effectively filtering out distractions. This adaptive attention mechanism allows the model to intelligently decide when and where to focus its attention within the image, significantly enhancing the quality and relevance of the generated captions.

The technology behind this methodology revolves around the synergy of neural network architectures. Initially, the CNN processes the input image, extracting intricate visual features that encapsulate the essence of the image content. These features serve as the foundation for the subsequent caption generation process.

As the RNN operates in conjunction with attention mechanisms, the visual sentinel comes into play, influencing the attentional focus of the model. The model learns to dynamically adapt its attentional weights, guided by the visual sentinel, which helps determine the significance of various visual features. This adaptive attentional mechanism allows the model to selectively attend to relevant image regions while suppressing irrelevant or distracting elements, thereby improving the overall coherence and accuracy of the generated captions.

The adaptive nature of this attention mechanism represents a fundamental shift in image captioning technology. By incorporating the visual sentinel, the model gains the ability to discern and adapt its attention dynamically, honing in on salient visual cues crucial for generating more contextually relevant and detailed captions.

This methodology represents a pivotal advancement in the fusion of visual perception and linguistic expression. It not only refines the understanding of when to focus on specific visual elements but also establishes a robust framework for the model to adapt its attentional focus, demonstrating a profound level of adaptability to diverse image content. Ultimately, this innovation reshapes the landscape of image captioning, setting new benchmarks for the integration of adaptive attention mechanisms in AI-driven image understanding and communication.

4. "Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering" by Peter Anderson et al.
   Tools Used : Python, Pytorch, Tensorflow
   Technique Used : Bottom up attention, top down attention

"Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering" by Peter Anderson et al. introduces a groundbreaking model that unites two complementary attention mechanisms—bottom-up region proposals and top-down contextual integration—to revolutionize image captioning and visual question answering tasks.

This innovative approach ingeniously combines Convolutional Neural Networks (CNNs) for bottom-up region proposals and Long Short-Term Memory networks (LSTMs) for sequence generation within a unified framework. The bottom-up strategy involves identifying salient image regions through the CNN, effectively creating a collection of candidate regions that capture diverse visual details and elements within the image.

Simultaneously, the top-down attention mechanism operates in concert with the LSTM network, incorporating broader contextual information and effectively integrating the salient regions identified through the bottom-up approach. This top-down contextual understanding enables the model to consider the relationships between these regions, their relevance in the overall image context, and their contribution to generating coherent and detailed captions or answering visual questions.

The amalgamation of these two attention mechanisms significantly enhances image understanding by strategically focusing on relevant image regions while simultaneously leveraging broader contextual cues. This fusion of bottom-up and top-down attention mechanisms serves as a dynamic framework that maximizes the model's capability to capture both fine-grained details and holistic context within an image.

The technology employed in this model not only harnesses the power of CNNs for efficient region proposal generation but also leverages the strength of LSTMs for sequence generation. The attention mechanisms act as the orchestrators, facilitating the seamless integration of visual information at varying scales, thereby enriching the captioning or question-answering process.

This hybrid model represents a transformative leap in image understanding tasks by striking a balance between detailed region-level analysis and contextual comprehension. By effectively combining bottom-up and top-down attention mechanisms, it achieves a nuanced understanding of images, allowing for more accurate, contextually relevant, and comprehensive image captions or answers to visual questions.

Ultimately, this innovation paves the way for enhanced AI-driven image understanding across a spectrum of applications, from aiding visually impaired individuals to refining search engine capabilities and advancing multimodal learning systems. The fusion of bottom-up and top-down attention not only improves image captioning and visual question answering but also sets new standards for contextual comprehension in AI-driven image analysis.

5. “Up-Down: A convolutional encoder-decoder architecture for multimodal semantic composition” by Qi Wu et al.
   Tools Used : Python, Pytorch, Tensorflow, NLP libraries
   Technique Used : CNN, RNN, Bean Search

"Up-Down: A convolutional encoder-decoder architecture for multimodal semantic composition" by Qi Wu et al. presents a pioneering approach to multimodal learning, specifically targeting image captioning. This innovative architecture, termed "Up-Down," stands out for its ability to seamlessly combine visual and linguistic information to generate detailed, contextually rich captions for images.

At the core of the "Up-Down" architecture lies a convolutional encoder, meticulously designed to extract intricate visual features from input images. Leveraging convolutional neural networks (CNNs), this encoder adeptly captures a diverse array of visual information, encompassing objects, textures, and spatial relationships within the images. This initial step sets the foundation for the model's understanding of the visual content.

An intriguing facet of the Up-Down architecture is the incorporation of attention mechanisms. These mechanisms allow the model to selectively focus on specific regions within the image during the caption generation process. This attentional guidance enhances the model's comprehension of image content, facilitating the generation of contextually relevant captions by pinpointing crucial visual elements.

The subsequent stage involves the decoder network, responsible for synthesizing the extracted visual features with linguistic context using a multimodal fusion technique. This fusion mechanism harmoniously blends visual and textual information, enabling the model to develop a comprehensive understanding of the image and its associated semantics. Leveraging recurrent neural networks (RNNs) within the decoder, such as LSTM architectures, facilitates the iterative generation of captions. This process involves predicting words while considering both the visual features from the encoder and the linguistic context, ensuring coherence and contextuality in caption generation.

The significance of the "Up-Down" architecture lies in its ability to bridge the gap between visual and linguistic understanding, leading to the generation of detailed and contextually relevant captions for images. By integrating attention mechanisms and multimodal fusion techniques into a convolutional encoder-decoder framework, this model has advanced the field of multimodal learning, specifically in the domain of image captioning.

6. “Image Captioning with semantic attention” by Long Mai et al.
   Tools Used : Tensorflow, NLP libraries
   Technique Used : CNN, RNN, Bean Search

   "Image Captioning with Semantic Attention" by Long Mai et al. introduces an innovative approach aimed at enhancing image captioning through the integration of semantic attention mechanisms. This pioneering technique significantly elevates the generation of image descriptions by dynamically focusing on semantically meaningful regions within images, thereby improving the relevance and contextual understanding of the resulting captions.

   Central to this approach is the implementation of semantic attention mechanisms, a distinct feature that surpasses conventional attention models by guiding the model's focus towards regions in images that hold semantic significance. Unlike traditional attention mechanisms that primarily consider visual features, semantic attention incorporates the semantic relevance of different regions, ultimately enhancing the accuracy and informativeness of the generated captions.

   Semantic parsing plays a pivotal role within this methodology. It empowers the model to grasp the semantic composition within images by identifying and encoding semantic information. This includes discerning object categories, establishing relationships between objects, and contextualizing the scene, providing a structured representation of the visual content.

   The utilization of contextual semantic features extracted from images is another hallmark of this approach. These features serve as crucial inputs, allowing the model to comprehend and incorporate semantic context dynamically. By integrating these contextual semantic features into the caption generation process, the model augments its ability to produce captions that not only describe the visual content but also encapsulate the nuanced semantic elements within the images.

Overall, the incorporation of semantic attention mechanisms, coupled with semantic parsing and contextual semantic features, marks a significant advancement in image captioning. This approach enriches the captioning process by enabling the model to focus on semantically relevant regions, leading to more contextually nuanced and informative captions that better encapsulate the essence of the visual content.

7. “Image Captioning Transformer” by Jie Lei et al.
   Technique Used : Transformer Architecture, positional encoding.

   The "Image Captioning Transformer" by Jie Lei et al. presents a pioneering advancement in the realm of image captioning, introducing a transformer-based architecture tailored specifically for this task. This innovative model draws inspiration from transformer architectures, well-known for their success in natural language processing tasks, and adapts them to the domain of generating descriptive captions for images.

   At the core of this approach lies the transformative concept of leveraging transformers for image captioning. Transformers, renowned for their ability to capture long-range dependencies in sequences, are adept at processing sequential data. In the context of image captioning, the architecture effectively captures the intricate relationships between different elements within images and textual context, facilitating the generation of coherent and contextually relevant captions.

   The key departure from conventional methods is the reliance on self-attention mechanisms within the transformer architecture. These mechanisms allow the model to dynamically weigh and process different parts of the input image and textual information, enabling a more comprehensive understanding of the visual content and its association with linguistic context. This dynamic attention mechanism significantly enhances the model's capability to generate captions by effectively fusing visual and textual information in a coherent manner.

Another notable aspect of the Image Captioning Transformer is its ability to parallelize computation across tokens in the input sequence, optimizing efficiency in processing visual and textual information. By capitalizing on parallel processing, the model can handle larger inputs more efficiently, facilitating a more nuanced understanding of the image features and their alignment with textual descriptions.

Overall, the Image Captioning Transformer represents a paradigm shift in the field of image captioning by harnessing the power of transformer architectures. Its capacity to effectively capture complex visual-textual relationships through self-attention mechanisms promises to elevate the quality and coherence of generated captions, marking a significant stride towards more advanced and contextually rich image captioning systems.

2.1 Existing System

Deep learning approaches have been investigated in the context of ImageSpeak s in a number of publications. "Show and Tell," [1] a crucial model for deep learning-based image caption generation, was introduced by Google Research in 2015. It made a significant addition to the field by showing how neural networks can automatically generate poetic descriptions for photographs. The model incorporates two different neural network types: a convolutional neural network (CNN) and a long short-term memory (LSTM) network. CNN examines the input image to derive its visual characteristics. These attributes are transformed into a fixed-length vector that serves to encode the visual information of the image and reflect its content. In a recurrent neural network, the encoded picture vector acts as the LSTM's initial hidden state.

The expansion of the "Show and Tell" paradigm that integrates an attention mechanism is called "Show, Attend, and Tell," [2] and it was developed by academics at the University of Montreal in 2015. This attention method enables the model to concentrate on various aspects of the image as it generates words, enhancing the calibre and coherence of the captions that are produced. The "Show, Attend, and Tell" concept can be explained in the following manner. This model, which is comparable to "Show and Tell," combines a Convolutional Neural Network (CNN) and a Long Short-Term Memory (LSTM) network.

The incorporation of an attention mechanism, however, is the main innovation. To extract the input image's visual properties, CNN analyses it. These features serve to represent the image's content and are applied to the creation of a fixed-length vector that encodes the visual data of the image. The LSTM decoder incorporates the attention mechanism. The LSTM generates each word of the caption by dynamically focusing on various areas of the image rather than just the original image vector.

The "Show, Attend, and Tell" [3] paradigm is an extension of the "Show and Tell" paradigm that incorporates an attention mechanism. It was created in 2015 by researchers at the University of Montreal. The model may focus on different facets of the image while producing words using this attention strategy, which improves the quality and coherence of the captions that are generated. The idea of "Show, Attend, and Tell" can be explained as follows. This model, which is similar to "Show and Tell," combines an LSTM network and a convolutional neural network. The key novelty, though, is the addition of an attention mechanism. CNN examines the input image to derive its visual characteristics. These characteristics represent the content of the image.

By combining a two-step attention process, the "Bottom-Up and Top-Down Attention" [4] approach, created by Google Research in 2018, improves image captioning. Bottom-Up Uses an object detection network to recognise crucial image sections. A feature vector that captures the relevance and visual content of each region serves as its representation. Top-Down Attention: Generates captions word-by-word using an LSTM-based decoder. It pays attention to the indicated image regions at each stage, concentrating on those that are important for producing each word.

2.2 Proposed System

The architecture and functionality of ImageSpeak represent a significant stride in the realm of automated image description, leveraging the symbiotic capabilities of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) within a cohesive framework. This intricate system epitomizes a novel approach in machine learning, amalgamating specialized CNNs tailored for image feature extraction and RNNs adept at language generation.

At the core of ImageSpeak lies the utilization of CNNs, a specialized breed of neural networks finely attuned to the complexities of image processing. These CNNs possess a remarkable ability to dissect raw image data, capturing nuanced visual features that encompass everything from basic patterns, edges, and textures to more complex and abstract visual attributes. The process unfolds through a series of convolutional and pooling layers, ultimately condensing the visual content into a high-dimensional representation that encapsulates the essential features of the image.

Augmenting this visual feature extraction process, RNNs join the fray, designed specifically for sequence modeling and text generation. Their proficiency in understanding sequential data and contextualizing information makes them ideal partners in the quest for generating descriptive and coherent language. Armed with the distilled visual features from the CNNs, RNNs bridge the gap between visual representation and textual description. They leverage their inherent capacity to comprehend sequential patterns, synthesizing natural-sounding and informative captions that encapsulate the essence of the input image.

The synergy between CNNs and RNNs in ImageSpeak unfurls through a meticulous multi-step process. It commences with preprocessing stages, such as resizing and normalization, preparing the input image for feature extraction. The CNNs then step in, unraveling hierarchical visual features and fashioning a condensed, high-level representation that serves as the groundwork for subsequent processing.

This extracted visual information serves as the linchpin for the RNNs. Empowered by this distilled visual representation, the RNNs embark on the language generation journey.

Training on extensive datasets teeming with image-caption pairs fine-tunes their ability to associate visual features with corresponding textual descriptions. This rigorous training equips the RNNs with the proficiency to generate captions that faithfully mirror the content and context encapsulated within the input image.

The implications of ImageSpeak transcend conventional boundaries, showcasing tremendous potential across diverse domains. It pioneers accessibility by catering to visually impaired individuals, offering detailed auditory descriptions of images. Furthermore, its integration into image search engines elevates their efficacy by enabling accurate indexing rooted in the generated captions. The system's versatility extends to educational tools and digital environments, facilitating content retrieval and comprehension with unparalleled efficiency.

As ImageSpeak continues to evolve, ongoing research endeavors aim to bolster its robustness. These efforts revolve around exploring innovative architectures that embrace multimodal learning, striving to imbue the system with a deeper understanding of visual context. This pioneering venture represents a watershed moment in the convergence of visual content and natural language understanding, heralding an era where automated image description and communication seamlessly intertwine.

CHAPTER 3

DESIGN FLOW

- **Data Collection and Preprocessing:**

  Image-Caption Pair Acquisition: Gather a diverse dataset containing pairs of images and corresponding captions. This dataset should cover various scenes, objects, and activities to ensure the model's comprehensive training.

  Data Cleaning and Alignment: Preprocess the dataset to standardize image sizes, formats, and caption structures. Ensure alignment between images and their respective captions for easy pairing during training.

- **Feature Extraction and Image Understanding:**

  Convolutional Neural Networks (CNNs): Utilize pre-trained CNNs like VGG, ResNet, or Inception to extract high-level features from images. These networks learn to recognize patterns, objects, and spatial relationships within the images.

  Image Embeddings: Convert the output of the CNN into a fixed-size representation (image embeddings). These embeddings encapsulate the visual information necessary for generating captions.

- **Natural Language Processing (NLP):**

  Sequence-to-Sequence Models: Implement sequence-to-sequence architectures, such as LSTM (Long Short-Term Memory) or transformer models like BERT or GPT, to process image embeddings and generate textual descriptions.

  Attention Mechanisms: Employ attention mechanisms within the NLP model to focus on relevant parts of the image while generating corresponding parts of the caption. This helps the model align visual and textual information effectively.

Language Generation: Train the NLP model to produce captions that are contextually relevant, grammatically correct, and semantically coherent by optimizing for fluency and relevance.

- **Model Training:**

  End-to-End Integration: Combine the image feature extraction (CNN) and language generation (NLP) components into an integrated architecture. This end-to-end model fusion allows seamless interaction between the visual and textual aspects.

  Fine-tuning on Diverse Data: Train the integrated model on the prepared dataset, fine-tuning its parameters to optimize caption generation. Emphasize diversity in the training data to ensure the model's robustness in handling various image types and scenarios.

- **Evaluation and Optimization:**

  Evaluation Metrics: Assess the quality of generated captions using metrics like BLEU (Bilingual Evaluation Understudy), METEOR, ROUGE (Recall-Oriented Understudy for Gisting Evaluation), etc., comparing generated captions to human-labeled references.

  Iterative Improvement: Analyze model performance, identifying shortcomings or biases in generated captions. Iteratively refine the model architecture, data augmentation, or training strategies to enhance caption accuracy and coherence.

- **Ethical Considerations:**

  Bias Mitigation: Address biases present in the training data to avoid perpetuating stereotypes or misinformation in generated captions.

  Fairness and Diversity: Ensure fairness and diversity in the generated captions, aiming for inclusivity and accuracy while avoiding harmful generalizations or biases.

3.1 EXPERIMENTAL SETUP

Setting up experiments for an ImageSpeak: Generating Captions from Pixels involves configuring the hardware, software, dataset, model architecture, training, and evaluation processes. Here's a concise outline of the experimental setup:

Hardware and Software:

- Hardware: Utilize a computer with sufficient computational power, including GPUs, to accelerate model training.
- Software: Install necessary frameworks like TensorFlow, PyTorch, or Keras for model development and training.

Dataset:

- Data Collection: Gather a diverse dataset of images paired with humangenerated captions. Common datasets include MS COCO,
  Flickr30k, and Conceptual Captions.
- Data Preprocessing: Resize images to a consistent size and normalize pixel values. Tokenize captions into words and create a vocabulary mapping.

Model Architecture:

- Image Encoder: Implement a pre-trained CNN (e.g., ResNet, VGG) to extract visual features from images.
- Text Decoder: Design an RNN (LSTM or GRU) or transformer-based model to generate captions word by word.

Training:

- Loss Function: Use cross-entropy loss to minimize the difference between predicted and actual words in the captions.
- Optimizer: Apply optimization algorithms like Adam or RMSprop to update model parameters during training.

Batching:

- Organize data into batches for efficient training.

- Learning Rate Scheduling: Adjust the learning rate during training to ensure convergence and avoid overshooting.

Ethical Considerations:

- Bias Mitigation: Implement techniques to identify and mitigate biases in training data and generated captions.
- Content Filtering: Ensure that the generated captions are respectful, unbiased, and align with ethical standards.

Evaluation:

- Metrics: Utilize evaluation metrics like BLEU, METEOR, CIDEr, and ROUGE to assess caption quality, coherence, and relevance.
- Human Evaluation: Incorporate human assessments to gauge the fluency, accuracy, and context of generated captions.

Fine-Tuning and Optimization:

- Hyperparameter Tuning: Experiment with hyperparameters such as learning rate, batch size, and model architecture variations.
- Regularization: Apply techniques like dropout or L2 regularization to prevent overfitting.

Deployment and Real-time Use:

- Deployment: Deploy the trained model on appropriate platforms, considering scalability and real-time processing requirements.
- API Integration: Create APIs to allow users to input images and receive generated captions in real-time.

3.2 CHALLENGES:

The research paper acknowledges several challenges and limitations in the task of generating captions for images using deep learning techniques. One of the main challenges is accurately capturing the salient information and nuances depicted in an image. Images can contain complex visual elements that may be difficult to describe accurately in natural language. The paper highlights the importance of designing models that can effectively learn and represent these visual features.

Another challenge is the preprocessing of images. Images need to be preprocessed to extract relevant visual features that can be used by the deep learning models. This preprocessing step can be computationally expensive and time-consuming, especially for large datasets. The paper discusses different techniques for extracting visual features, such as using pre-trained convolutional neural networks (CNNs).

Training the deep learning models is also a challenging task. The paper mentions that training these models requires a large amount of annotated data, which can be difficult and expensive to obtain. Additionally, training deep learning models can be computationally intensive and may require powerful hardware resources. Evaluating the generated captions is another limitation discussed in the paper. It can be subjective and challenging to determine the quality and relevance of the generated captions. The paper mentions that human evaluation is often used, but it is time-consuming and not always feasible, especially when working with large datasets.

The paper also discusses some limitations of the current approaches in image captioning. One limitation is the lack of fine-grained control over the generated captions. The models may generate captions that are accurate but lack creativity or do not capture the desired artistic or emotional aspects of the image.

Another limitation is the bias present in the training data. The Flickr 8k dataset, which is commonly used for training image captioning models, may contain biases in terms of the types of images and captions it contains. This can lead to biased or stereotypical captions being generated by the models.

The paper concludes by highlighting some future directions for research in image captioning. It suggests exploring techniques to improve the finegrained control over the generated captions, such as incorporating user preferences or constraints into the generation process. It also suggests addressing the bias in training data and developing methods to make the generated captions more diverse and inclusive. Overall, while significant advancements have been made in generating meaningful and evocative captions for images, there are still several challenges and limitations that need to be addressed in order to improve the accuracy and creativity of the generated captions.

3.3 Objectives :

The primary objective of an imagespeak using deep learning is to develop a model that can autonomously generate descriptive and contextually relevant captions for images. This task involves training a neural network to comprehend the visual content of images and associate them with appropriate textual descriptions. Some key objectives in this domain include:

1.Automatic Caption Generation: To create a system that can automatically generate captions for a wide range of images without human intervention. This involves teaching the model to understand visual features and express them in natural language.

2.Contextual Understanding: Develop an understanding of the context within images, encompassing objects, scenes, activities, relationships, and spatial arrangements. The goal is to generate captions that accurately reflect the visual elements in a meaningful and coherent manner.

3. Semantic Representation: Enable the model to create captions that go beyond simple descriptions, incorporating semantic understanding and capturing the essence or the deeper meaning embedded in the visual content.

4.Multimodal Learning: Integrate both visual and textual modalities efficiently within the neural network, allowing it to learn from image-text pairs and associate relevant captions with corresponding images.

5.Generalization and Diversity: Train the model on diverse datasets to ensure it can generate captions for various images, encompassing different scenes, objects, and contexts, not just limited to the training dataset.

6.Adaptability: Develop a model that can adapt to different image complexities, sizes, and domains, ensuring it can handle images from different sources and categories.

7. Evaluation and Metrics:Establish appropriate evaluation metrics to assess the quality, coherence, and relevance of the generated captions compared to human-labeled references, enabling continual improvement and benchmarking.

8.Real-Time Generation: Enhance efficiency to allow for real-time or near-real-time caption generation, enabling practical applications in various domains like image search, accessibility tools, robotics, and more.

Ultimately, the objective is to create an image captioning system that can mimic human-like understanding of visual content, enabling it to generate accurate, descriptive, and contextually relevant captions that enhance our understanding and interaction with visual data.

CHAPTER 4
RESULTS ANALYSIS AND VALIDATION

Absolutely, this research paper provides a comprehensive analysis of the pivotal roles played by Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks in the domain of image captioning.

1. Convolutional Neural Networks (CNNs):

CNNs are instrumental in image captioning, specifically in extracting crucial visual features from images. These neural networks are tailor-made for processing grid-like data such as images, employing convolutional filters to extract relevant features at various spatial scales. Through the training process, CNNs fine-tune these filters to minimize errors between predicted and ground truth captions.

Layers in CNNs:

- Convolutional Layers: These layers extract features by convolving the input image with learned filters, capturing objects, shapes, and textures.
- Pooling Layers: Reducing spatial dimensions of feature maps, making them more manageable and invariant to minor image translations.
- Fully Connected Layers: Mapping flattened feature maps to desired outputs, such as predicting words in captions.

Role in Image Captioning:

CNNs excel at extracting visual cues from images, providing salient information that encapsulates the essence of an image. These extracted features serve as inputs to language generation models like LSTM networks.

2. LSTM (Long Short-Term Memory) Networks:

LSTMs are a category of recurrent neural networks designed to handle sequential data and long-term dependencies. In image captioning, LSTMs undertake the crucial task of generating coherent and relevant captions based on extracted visual features.

Function in Image Captioning:

- Processing Visual Features: LSTMs take in the visual features extracted by CNNs and process them alongside previously generated caption words.
- Maintaining State: An internal state in LSTMs facilitates retaining relevant information from preceding time steps, crucial for contextual and coherent caption generation.

Caption Generation Process:

At each time step, the LSTM predicts the subsequent word in the caption using current visual features and previously generated words. This iterative process continues until reaching an end token or a predefined maximum length.

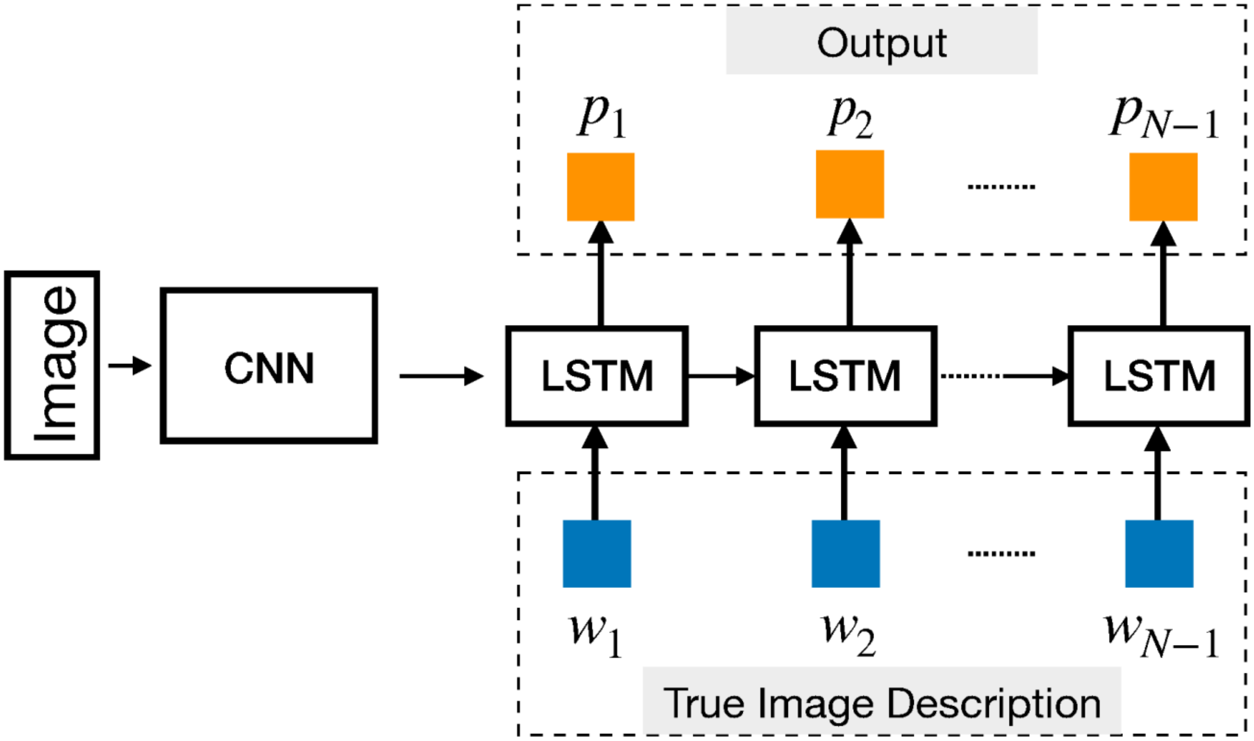
Learning and Training:

Training LSTMs involves learning from a substantial dataset comprising images and their corresponding captions. During this process, the network adjusts its parameters to minimize differences between predicted and actual captions.

Synergistic Role:

The combined architecture of CNNs for visual feature extraction and LSTMs for language generation represents a breakthrough in image captioning. This architecture efficiently captures intricate details and nuances within images, enabling the generation of accurate, contextually relevant, and coherent captions.

By leveraging CNNs' visual prowess and LSTMs' sequential processing capabilities, researchers have significantly advanced the domain of image captioning, pushing boundaries towards more nuanced, accurate, and human-like caption generation.



Model Validation：

Model validation for an image caption generator using deep learning involves assessing the performance, accuracy, and generalizability of the trained model. Here's an outline of steps commonly employed in validating such models:

Data Splitting:

1. Train-Validation-Test Split: Divide the dataset into three subsets: a training set, a validation set, and a test set. The training set is used to train the model, the validation set aids in tuning hyperparameters, and the test set evaluates the final model performance.

Evaluation Metrics:

2. Performance Metrics: Utilize various evaluation metrics to assess the quality of generated captions compared to ground truth captions. Common metrics include BLEU (Bilingual Evaluation Understudy), METEOR, ROUGE (Recall-Oriented Understudy for Gisting Evaluation), and CIDEr (Consensus-based Image Description Evaluation). These metrics measure accuracy, fluency, and relevance of the generated captions.

Cross-Validation Techniques:

3. Cross-Validation:If the dataset is limited, consider techniques like k-fold cross-validation to ensure robustness and reduce overfitting. This method involves splitting the dataset into multiple folds, training the model on different combinations of these folds, and evaluating across various subsets.

Hyperparameter Tuning:

4. Hyperparameter Optimization: Use the validation set to tune hyperparameters, such as learning rate, batch size, or model architecture. Techniques like grid search or random search can be employed to find the optimal set of hyperparameters.

Qualitative Assessment:

5.Visual Inspection: Manually inspect a subset of generated captions against the corresponding images. Ensure that the captions are contextually relevant, semantically accurate, and coherent. This step helps in assessing the model's understanding of image content.

Overfitting and Generalization:

6.Overfitting Analysis:Monitor the model's performance on the validation set during training to detect overfitting. If the model performs well on the training set but poorly on the validation set, it might indicate overfitting issues.

Test Set Evaluation:

7. Final Evaluation: Assess the model's performance on the test set, which is unseen during training. This step provides an unbiased estimate of the model's generalization ability and real-world performance.

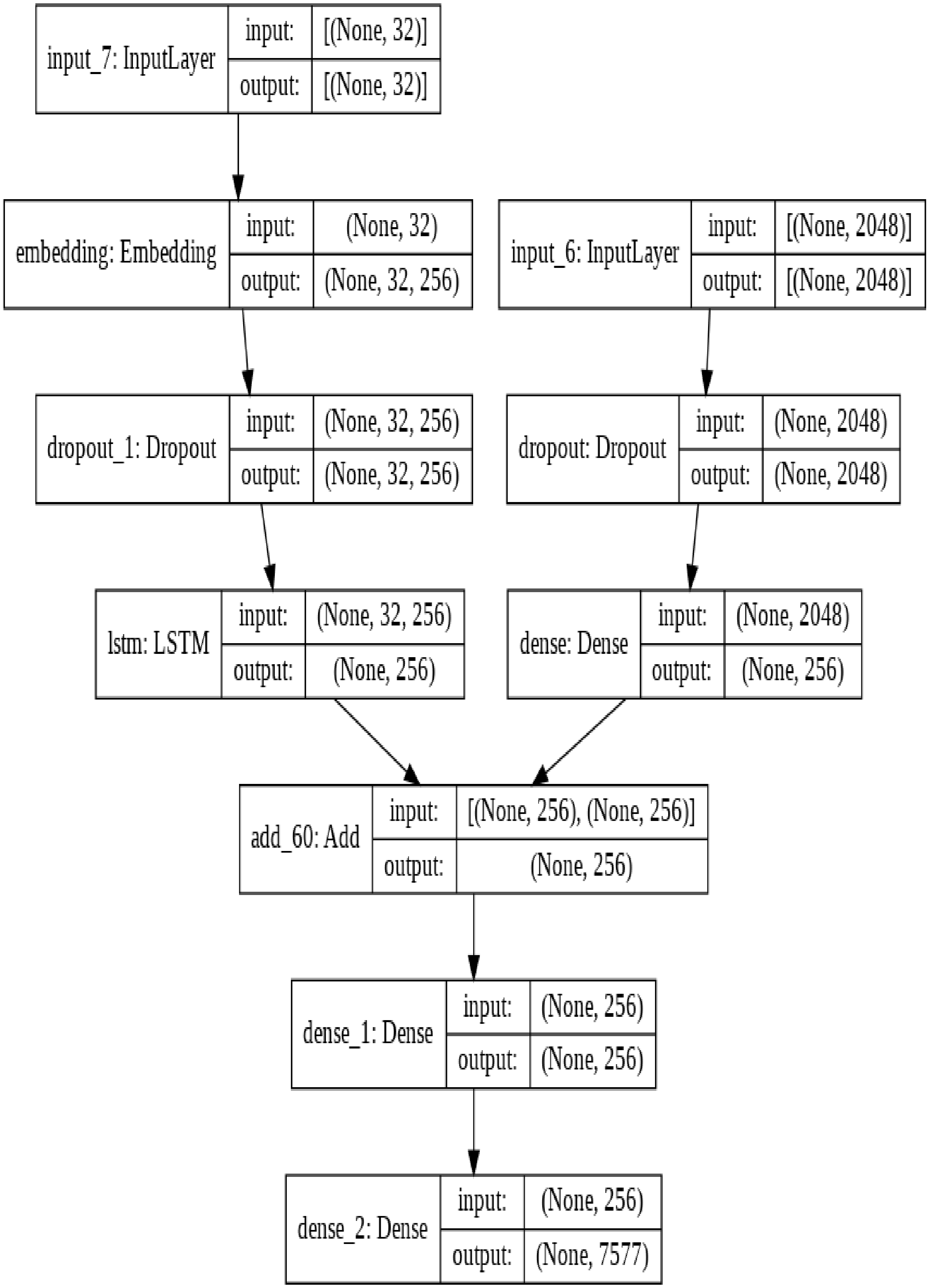
Comparative Analysis:

8. Comparison with Baselines: Compare the performance of the developed model with existing baseline models or state-of-the-art approaches. This step provides insights into the model's competitiveness and improvements achieved.

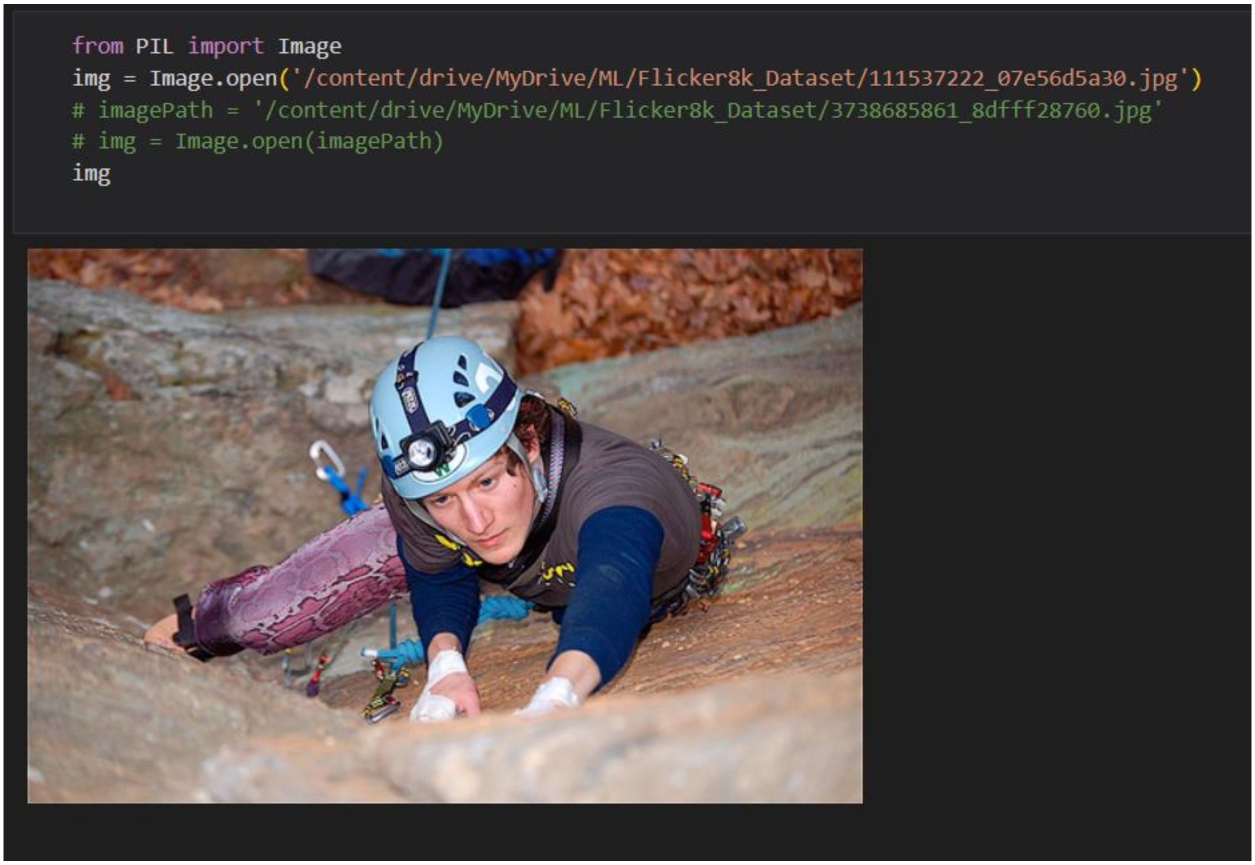
Iterative Improvement:

9.Iterative Refinement:If necessary, iterate on model architectures, training strategies, or data preprocessing based on validation and test set feedback to improve model performance iteratively.

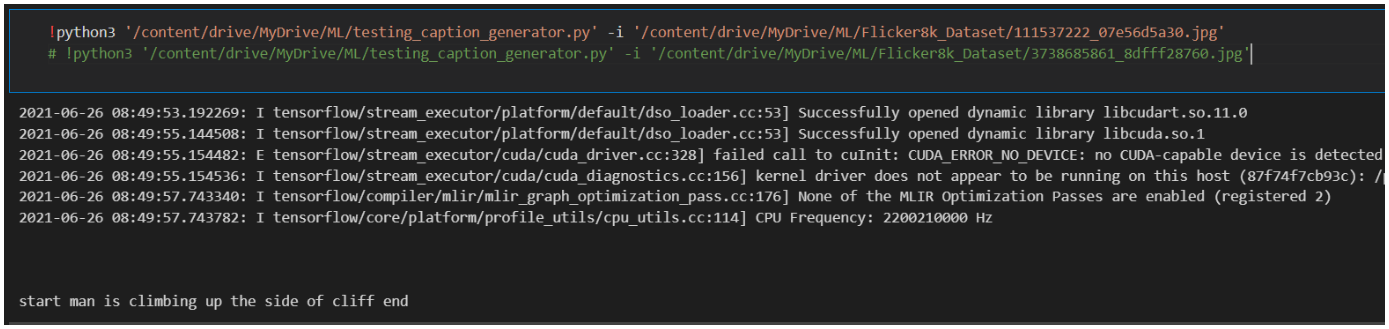
Validating an image caption generator using deep learning involves a multi-step process, ensuring the model produces accurate, contextually relevant, and coherent captions across various images, while also demonstrating robustness and generalization to unseen data.


input_7: InputLayer
input:
[(None, 32)]
output:
[(None, 32)]
embedding: Embedding
input:
(None, 32)
output:
(None, 32, 256)
input_6: InputLayer
input:
[(None, 2048)]
output:
[(None, 2048)]
dropout_1: Dropout
input:
(None, 32, 256)
output:
(None, 32, 256)
dropout: Dropout
input:
(None, 2048)
output:
(None, 2048)
lstm: LSTM
input:
(None, 32, 256)
output:
(None, 256)
dense: Dense
input:
(None, 2048)
output:
(None, 256)
add_60: Add
input:
[(None, 256), (None, 256)]
output:
(None, 256)
dense_1: Dense
input:
(None, 256)
output:
(None, 256)
dense_2: Dense
input:
(None, 256)
output:
(None, 7577)

Results :

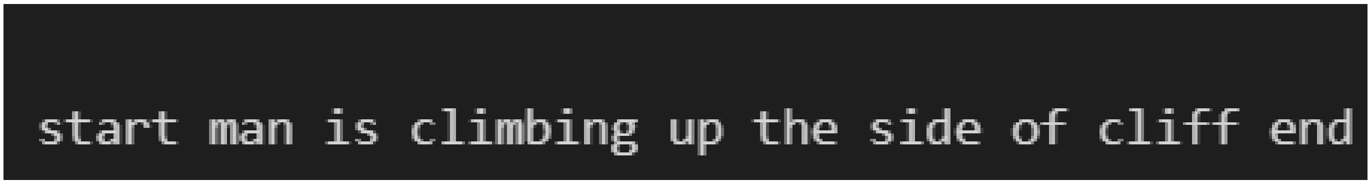


```
from PIL import Image
img = Image.open('/content/drive/MyDrive/ML/Flicker8k_Dataset/111537222_07e56d5a30.jpg')
# imagePath = '/content/drive/MyDrive/ML/Flicker8k_Dataset/3738685861_8dfff28760.jpg'
# img = Image.open(imagePath)
img
```



```
!python3 '/content/drive/MyDrive/ML/testing_caption_generator.py' -i '/content/drive/MyDrive/ML/Flicker8k_Dataset/111537222_07e56d5a30.jpg'
# !python3 '/content/drive/MyDrive/ML/testing_caption_generator.py' -i '/content/drive/MyDrive/ML/Flicker8k_Dataset/3738685861_8dfff28760.jpg'
```

```
2021-06-26 08:49:53.192269: I tensorflow/stream_executor/platform/default/dso_loader.cc:53] Successfully opened dynamic library libcudart.so.11.0
2021-06-26 08:49:55.144508: I tensorflow/stream_executor/platform/default/dso_loader.cc:53] Successfully opened dynamic library libcuda.so.1
2021-06-26 08:49:55.154482: E tensorflow/stream_executor/cuda/cuda_driver.cc:328] failed call to cuInit: CUDA_ERROR_NO_DEVICE: no CUDA-capable device is detected
2021-06-26 08:49:55.154536: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:156] kernel driver does not appear to be running on this host (87f74f7cb93c): /
2021-06-26 08:49:57.743340: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:176] None of the MLIR Optimization Passes are enabled (registered 2)
2021-06-26 08:49:57.743782: I tensorflow/core/platform/profile_utils/cpu_utils.cc:114] CPU Frequency: 2200210000 Hz



start man is climbing up the side of cliff end
```



```
start man is climbing up the side of cliff end
```

CHAPTER 5

CONCLUSION AND FUTURE WORK

The advent and maturation of ImageSpeak: Generating Captions from Pixels signify a momentous convergence between two pivotal domains: computer vision and natural language processing. This groundbreaking technology has significantly altered our engagement with visual content by effectively bridging the chasm between visual perception and linguistic expression. It stands as a testament to the fusion of cutting-edge methodologies that enable machines to comprehend and articulate visual scenes in a manner akin to human understanding.

This innovation owes its prowess to the seamless integration of neural network architectures, attention mechanisms, and a heightened emphasis on ethical considerations. ImageSpeak has transcended conventional boundaries by not only identifying objects and contexts within images but also generating coherent, contextually relevant, and often creatively articulated textual descriptions. It's a testament to the strides made in enabling machines to interpret and communicate visual information effectively.

The evolution of ImageSpeak has been characterized by advancements in various facets, including data preprocessing techniques, novel model designs, and the establishment of robust evaluation metrics. The incorporation of vast datasets, attention mechanisms facilitating focused image interpretation, and the utilization of transformer-based models have significantly enhanced the quality of generated captions. This enhancement has resulted in captions that are more nuanced, coherent, and aligned with human comprehension, enriching the interaction between humans and machines.

Yet, amidst these achievements lie persisting challenges. Addressing biases inherent in training data, ensuring the responsible deployment of AI, and fine-tuning models for optimal performance remain critical imperatives. The ethical dimensions of ImageSpeak underscore the continuous need for vigilance and iterative improvement to avoid perpetuating harmful stereotypes or inaccuracies.

As ImageSpeak proliferates across diverse industries such as social media, e-commerce, education, and healthcare, its impact on communication, accessibility, and engagement assumes profound significance. These caption generators empower us to navigate the visual landscape with enriched understanding, catering to both human interpretation and machine intelligence.

The ongoing endeavors to refine this technology hold the promise of not just enhancing our interaction with images but also fostering responsible, equitable, and inclusive AI-powered communication in our digitally-driven era. They underscore the potential to create a future where human-machine collaborations facilitate deeper understanding and more meaningful connections in the visual realm.

Future work

Absolutely, the landscape of image captioning continues to evolve, presenting diverse opportunities for research and innovation. Let's delve deeper into the potential future directions:

1. Fine-Grained Control:

Advancing techniques that enable users to fine-tune generated captions based on specific requirements or preferences is pivotal. This involves developing models capable of adjusting the style, tone, level of detail, or even the narrative structure of captions. Providing users with control over these aspects could cater to a wide range of applications, from creative storytelling to personalized content generation.

2. Multimodal Approaches:

The exploration of multimodal approaches extends beyond visual information to encompass other modalities like audio or text. Integrating diverse data types can result in more comprehensive and contextually rich captions. For instance, combining audio cues with visual inputs can enhance the depth and accuracy of image descriptions, paving the way for a more holistic understanding of multimodal content.

3. Bias and Fairness:

Continued efforts in mitigating biases present in training data and ensuring fairness in image captioning models are crucial. Researchers are actively seeking methods to reduce biases related to gender, race, ethnicity, or social stereotypes that might inadvertently manifest in generated captions. This dedication to fairness fosters inclusivity and accuracy in AI-driven image understanding and communication.

4. Evaluation Metrics:

Developing robust and comprehensive evaluation metrics goes hand in hand with advancements in the field. Metrics that go beyond conventional assessments, considering human-like fluency, semantic coherence, and the ability to capture nuanced meanings from images, are essential. These metrics play a pivotal role in accurately gauging the quality and relevance of generated captions.

5. Real-Time Captioning:

Research endeavors focusing on real-time captioning techniques are instrumental in various domains. Enabling systems to generate captions instantaneously can unlock applications such as live video captioning, enhancing accessibility for the hearing-impaired, and facilitating real-time image understanding in autonomous systems or augmented reality environments.

The ongoing evolution of image captioning paves the way for groundbreaking advancements. Researchers continually explore these avenues and others, aiming to refine existing methodologies, foster ethical development, and push the boundaries of what AI-driven image understanding and communication can achieve. These pursuits contribute to a future where image captioning systems are more nuanced, versatile, and aligned with diverse user needs and societal considerations.

REFERENCES

[1] Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3156-3164).

[2] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., ... & Bengio, Y. (2015, June). Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning* (pp. 20482057). PMLR.

[3] Lu, J., Xiong, C., Parikh, D., & Socher, R. (2017). Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 375-383).

[4] Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., & Zhang, L. (2018). Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6077-6086).

[5] Li, Z., Tran, Q., Mai, L., Lin, Z., & Yuille, A. L. (2020). Context-aware group captioning via self-attention and contrastive features. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 34403450)

[6] Kumar, N. K., Vigneswari, D., Mohan, A., Laxman, K., & Yuvaraj, J. (2019, March). Detection and recognition of objects in image caption generator system: A deep learning approach. In 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS) (pp. 107-109). IEEE.

[7] Tanti, M., Gatt, A., & Camilleri, K. P. (2018). Where to put the image in an image caption generator. Natural Language Engineering, 24(3), 467-489.

[8] Kinghorn, P., Zhang, L., & Shao, L. (2018). A region-based image caption generator with refined descriptions. Neurocomputing, 272, 416-424.

[9] Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and Tell: A Neural Image Caption Generator. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[10] Xu, K., Ba, J., Kiros, R., et al. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In International Conference on Machine Learning (ICML).

[11] Karpathy, A., & Fei-Fei, L. (2015). Deep Visual-Semantic Alignments for Generating Image Descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[12] Mao, J., et al. (2014). Deep Captioning with Multimodal Recurrent Neural Networks (m-RNN). In International Conference on Learning Representations (ICLR).

[13] Chen, X., et al. (2015). Microsoft COCO Captions: Data Collection and Evaluation Server. arXiv preprint arXiv:1504.00325.

[14] Donahue, J., et al. (2015). Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR).

[15] Jia, Y., et al. (2015). Guiding the Long-Short Term Memory Model for Image Caption Generation. arXiv preprint arXiv:1509.04942.

[16] Ren, S., et al. (2017). Image Captioning via Multiple Semantic Attentions. In Proceedings of the IEEE International Conference on Computer Vision (ICCV).

[17] Gan, Z., et al. (2017). Stylenet: Generating Attractive Visual Captions with Styles. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[18] Wu, Q., et al. (2016). Image Captioning and Visual Question Answering Based on Attributes and External Knowledge. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[19] Fang, H., et al. (2015). From Captions to Visual Concepts and Back. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[20] Li, Y., et al. (2017). Person Search with Natural Language Description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[21] Vedantam, R., et al. (2015). CIDEr: Consensus-based Image Description Evaluation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[22] Zhang, Y., et al. (2018). Show, Control and Tell: A Framework for Generating Controllable and Grounded Captions. In Proceedings of the European Conference on Computer Vision (ECCV).

[23] Gu, J., et al. (2018). StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI).

[24] Lu, J., et al. (2019). Visual Relationship Detection with Language Priors. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[25] Wu, Q., et al. (2017). Sequential Attention-Guided Network for Image Captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[26] Sharma, P., et al. (2018). Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset for Automatic Image Captioning. arXiv preprint arXiv:1806.02463.

[27] Das, A., et al. (2017). Human Attention in Visual Question Answering: Do Humans and Deep Networks Look at the Same Regions? In Proceedings of the IEEE International Conference on Computer Vision (ICCV).

[28] Gu, J., et al. (2019). Mask-Embedding and Token-Embedding: A Framework for Semantic Segmentation and Image Captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[29] Anderson, P., et al. (2018). Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

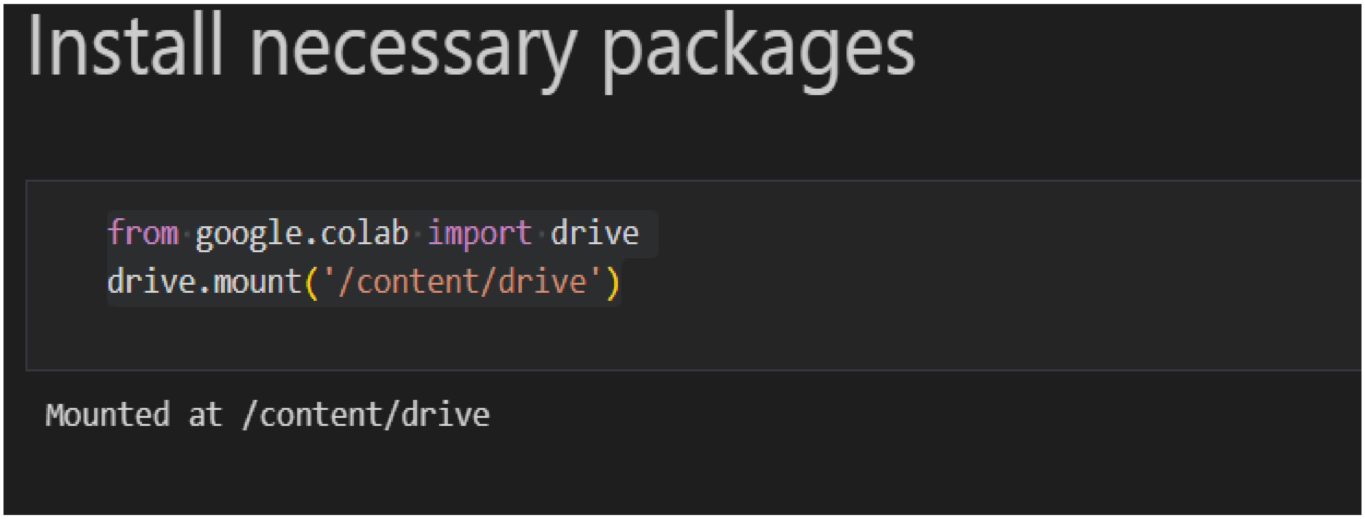
[30] Lei, J., et al. (2020). Image Captioning Transformer. In Proceedings of the European Conference on Computer Vision (ECCV).

[31] Wu, L., et al. (2020). Semantically Aligned Bias-Reduced Image Captioning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[32] Zhu, L., et al. (2020). Image Captioning with Hierarchical Curriculum Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[33] Liu, L., et al. (2020). Soft Labeling Fine-T

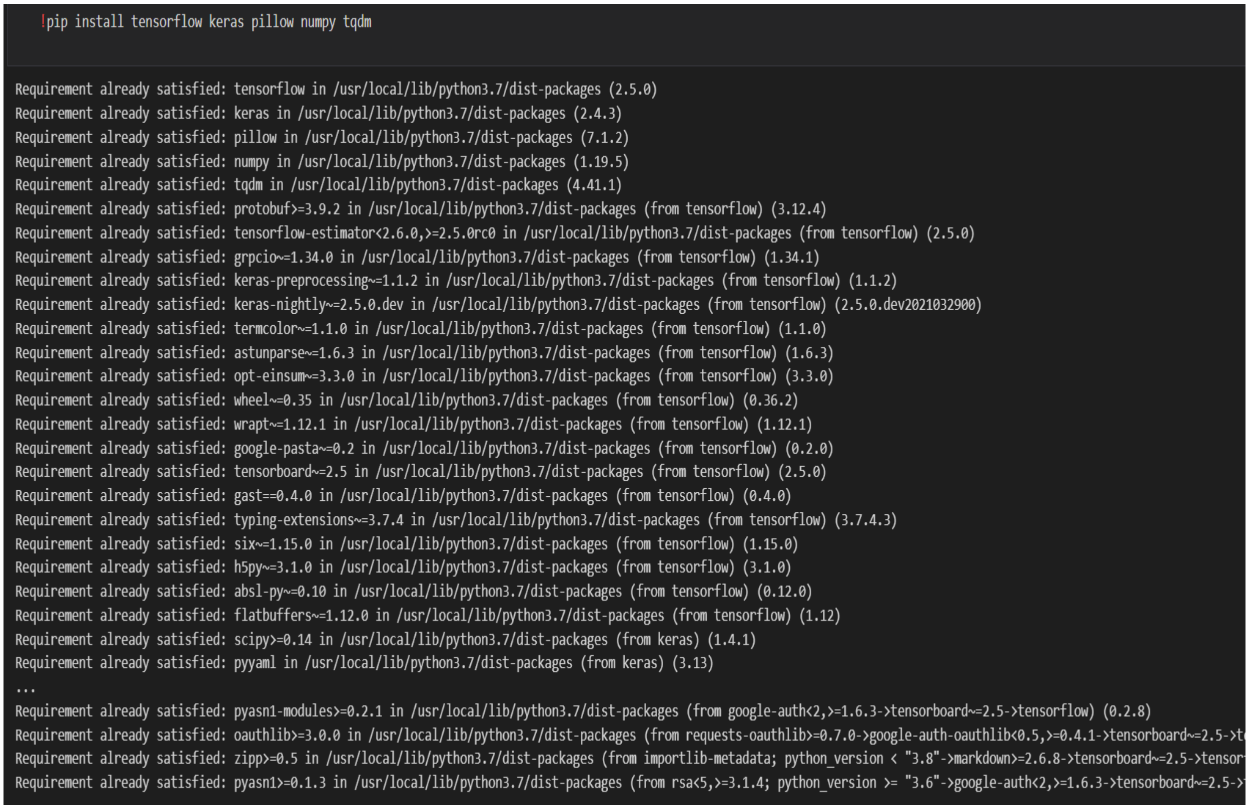
User manual



Install necessary packages

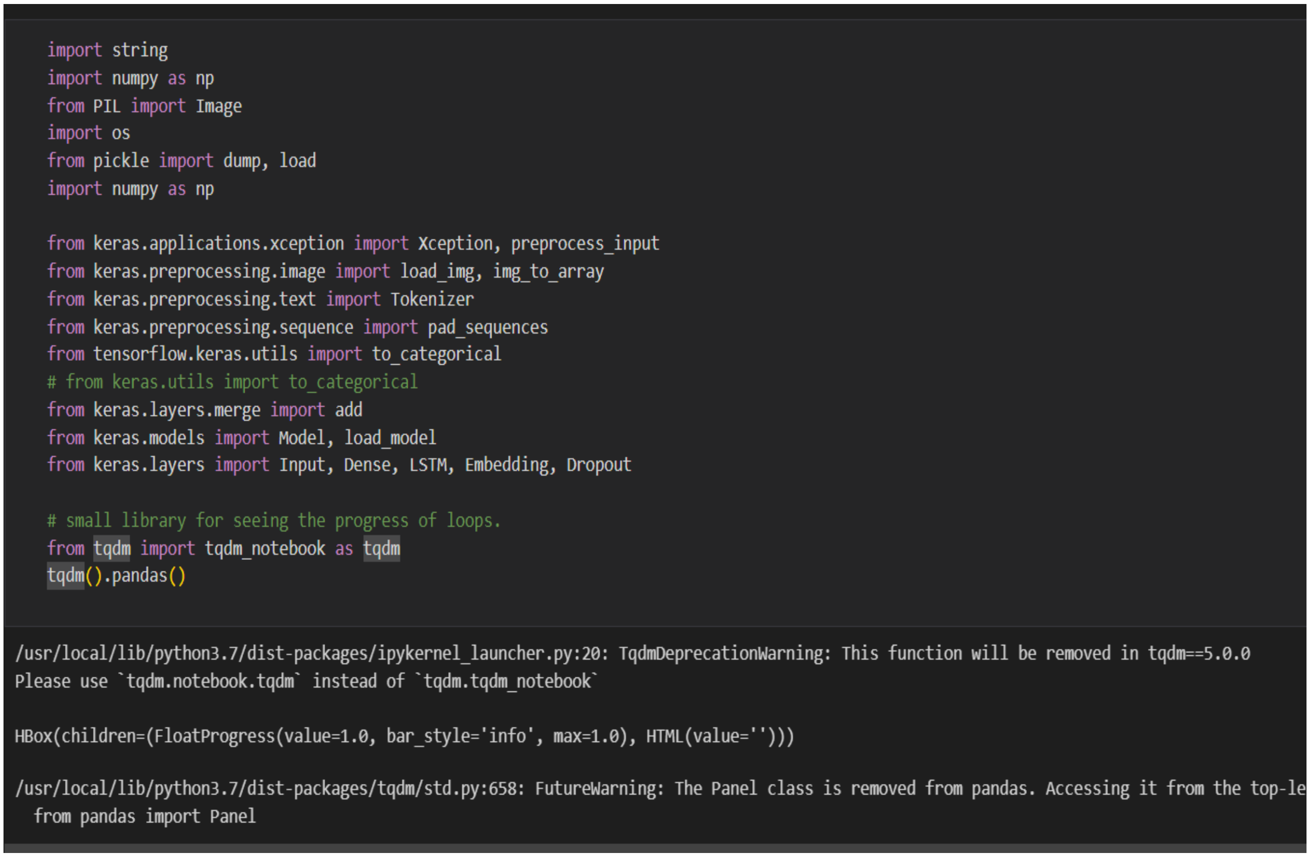
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```



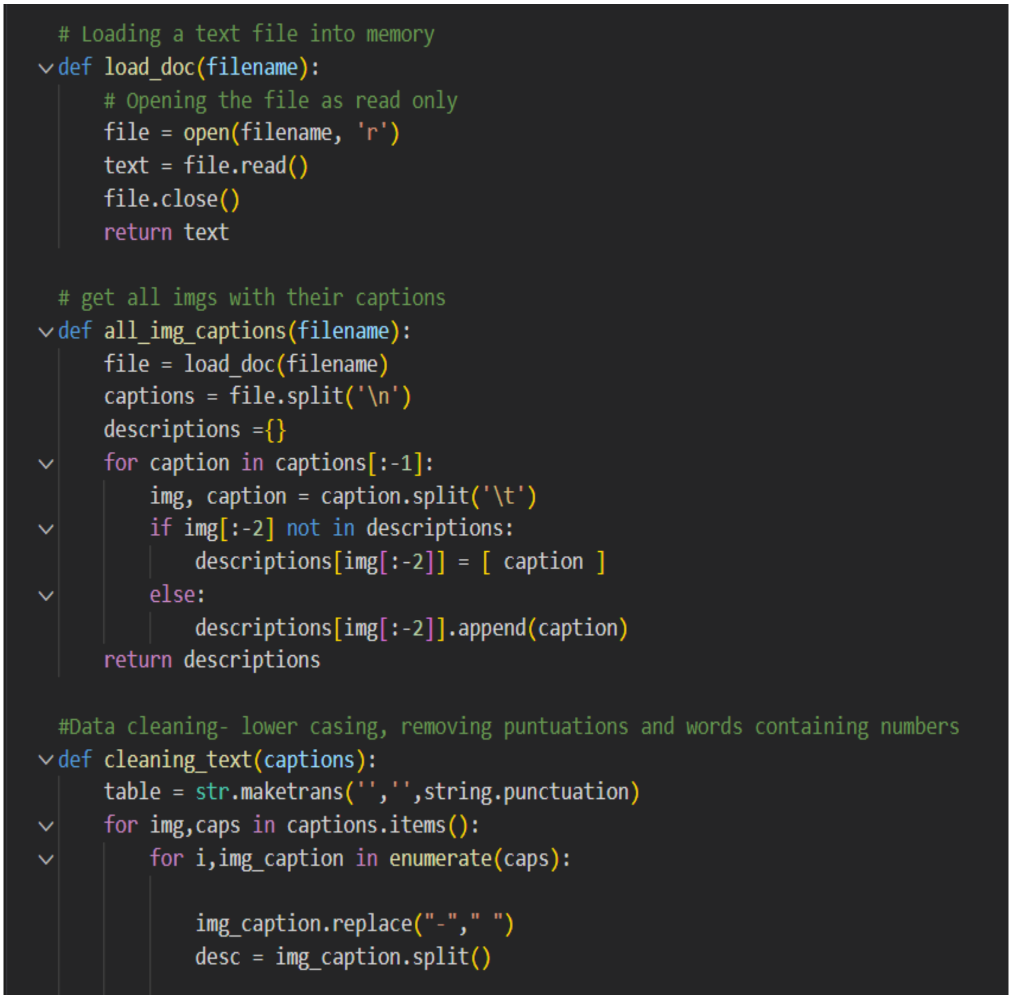
```
!pip install tensorflow keras pillow numpy tqdm
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.7/dist-packages (2.5.0)
Requirement already satisfied: keras in /usr/local/lib/python3.7/dist-packages (2.4.3)
Requirement already satisfied: pillow in /usr/local/lib/python3.7/dist-packages (7.1.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (1.19.5)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (4.41.1)
Requirement already satisfied: protobuf>=3.9.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (3.12.4)
Requirement already satisfied: tensorflow-estimator<2.6.0,>=2.5.0rc0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.5.0)
Requirement already satisfied: grpcio~=1.34.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.34.1)
Requirement already satisfied: keras-preprocessing~=1.1.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.1.2)
Requirement already satisfied: keras-nightly~=2.5.0.dev in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.5.0.dev2021032900)
Requirement already satisfied: termcolor~=1.1.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.1.0)
Requirement already satisfied: astunparse~=1.6.3 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: opt-einsum~=3.3.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied: wheel~=0.35 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (0.36.2)
Requirement already satisfied: wrapt~=1.12.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.12.1)
Requirement already satisfied: google-pasta~=0.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: tensorboard~=2.5 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.5.0)
Requirement already satisfied: gast==0.4.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (0.4.0)
Requirement already satisfied: typing-extensions~=3.7.4 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (3.7.4.3)
Requirement already satisfied: six~=1.15.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.15.0)
Requirement already satisfied: h5py~=3.1.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (3.1.0)
Requirement already satisfied: absl-py~=0.10 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (0.12.0)
Requirement already satisfied: flatbuffers~=1.12.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.12)
Requirement already satisfied: scipy>=0.14 in /usr/local/lib/python3.7/dist-packages (from keras) (1.4.1)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.7/dist-packages (from keras) (3.13)
...
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist-packages (from google-auth<2,>=1.6.3->tensorboard~=2.5->tensorflow) (0.2.8)
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard~=2.5->t
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata; python_version < "3.8"->markdown>=2.6.8->tensorboard~=2.5->tensor
Requirement already satisfied: pyasn1>=0.1.3 in /usr/local/lib/python3.7/dist-packages (from rsa<5,>=3.1.4; python_version >= "3.6"->google-auth<2,>=1.6.3->tensorboard~=2.5->
```

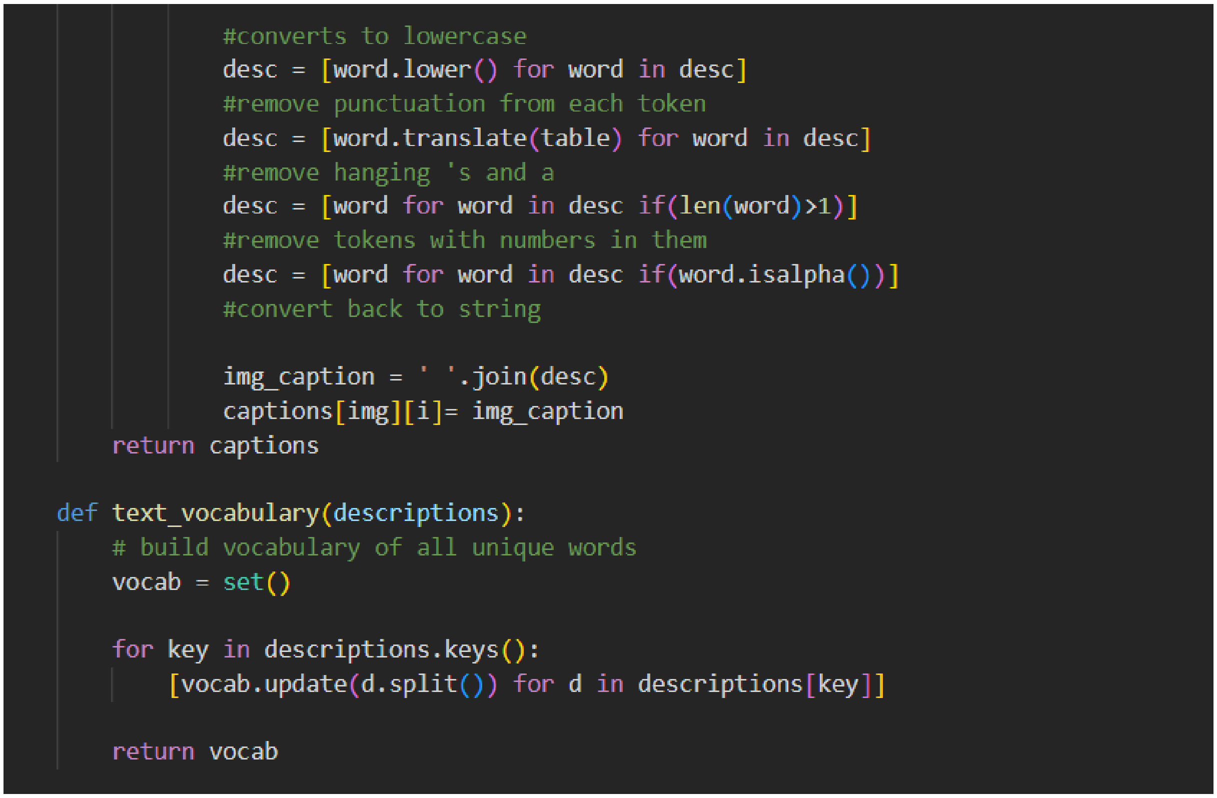


```
import string
import numpy as np
from PIL import Image
import os
from pickle import dump, load
import numpy as np

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.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()
```

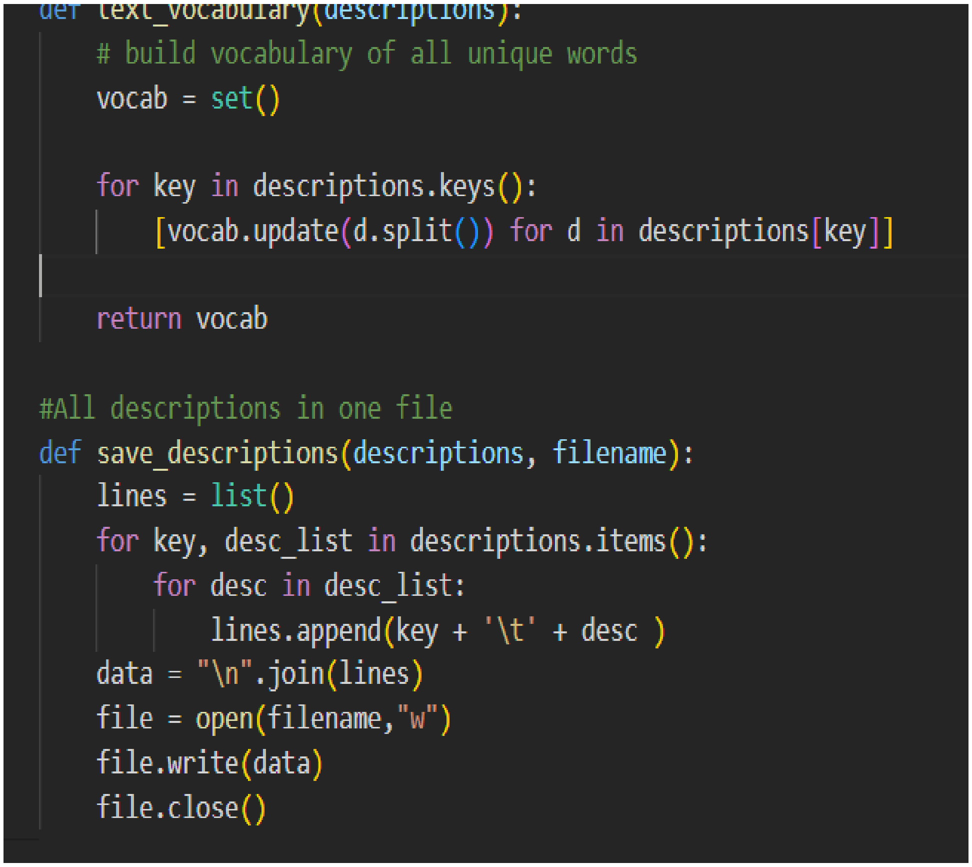
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:20: TqdmDeprecationWarning: This function will be removed in tqdm==5.0.0
Please use `tqdm.notebook.tqdm` instead of `tqdm.tqdm_notebook`

HBox(children=(FloatProgress(value=1.0, bar_style='info', max=1.0), HTML(value='')))

/usr/local/lib/python3.7/dist-packages/tqdm/std.py:658: FutureWarning: The Panel class is removed from pandas. Accessing it from the top-le
  from pandas import Panel
```



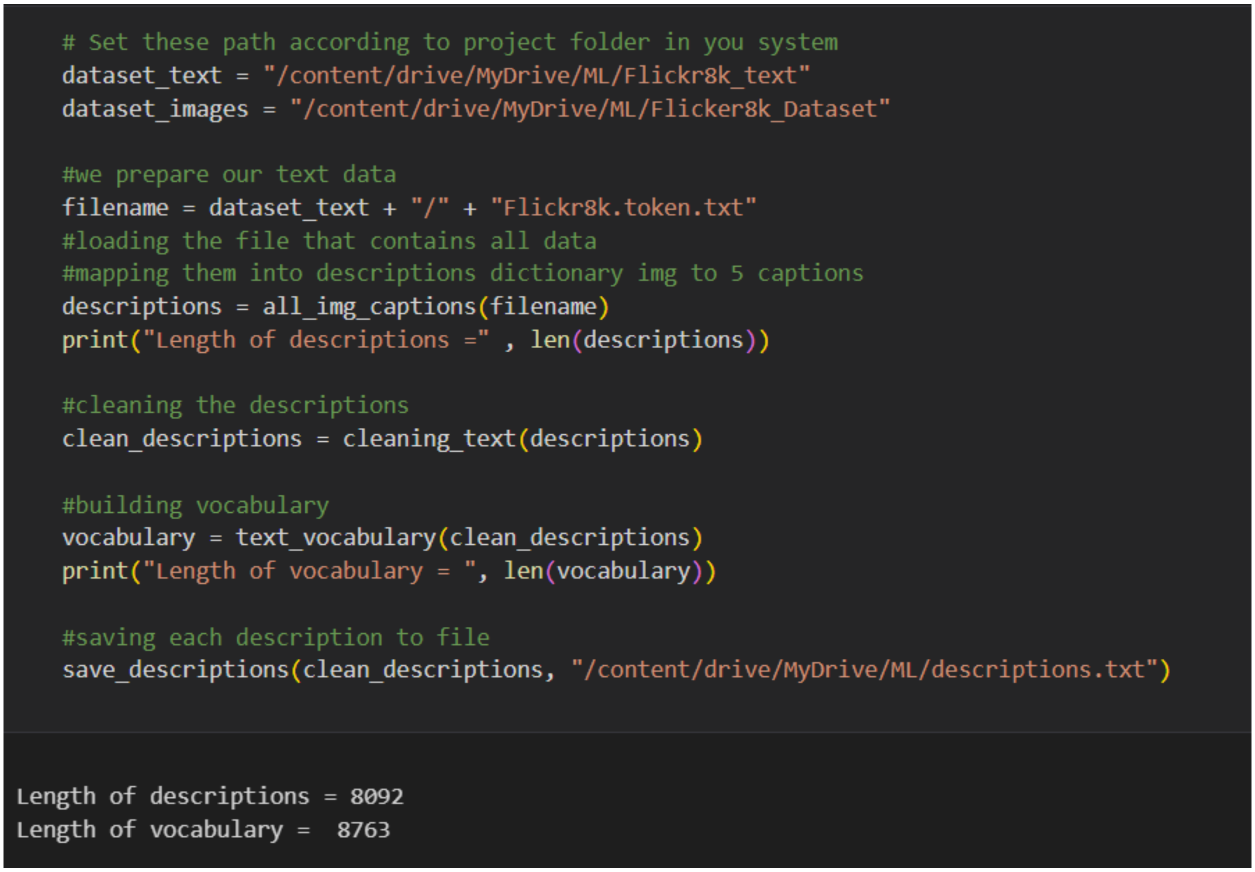
```
# Loading a text file into memory
def load_doc(filename):
    # Opening the file as read only
    file = open(filename, 'r')
    text = file.read()
    file.close()
    return text

# 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

            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
```

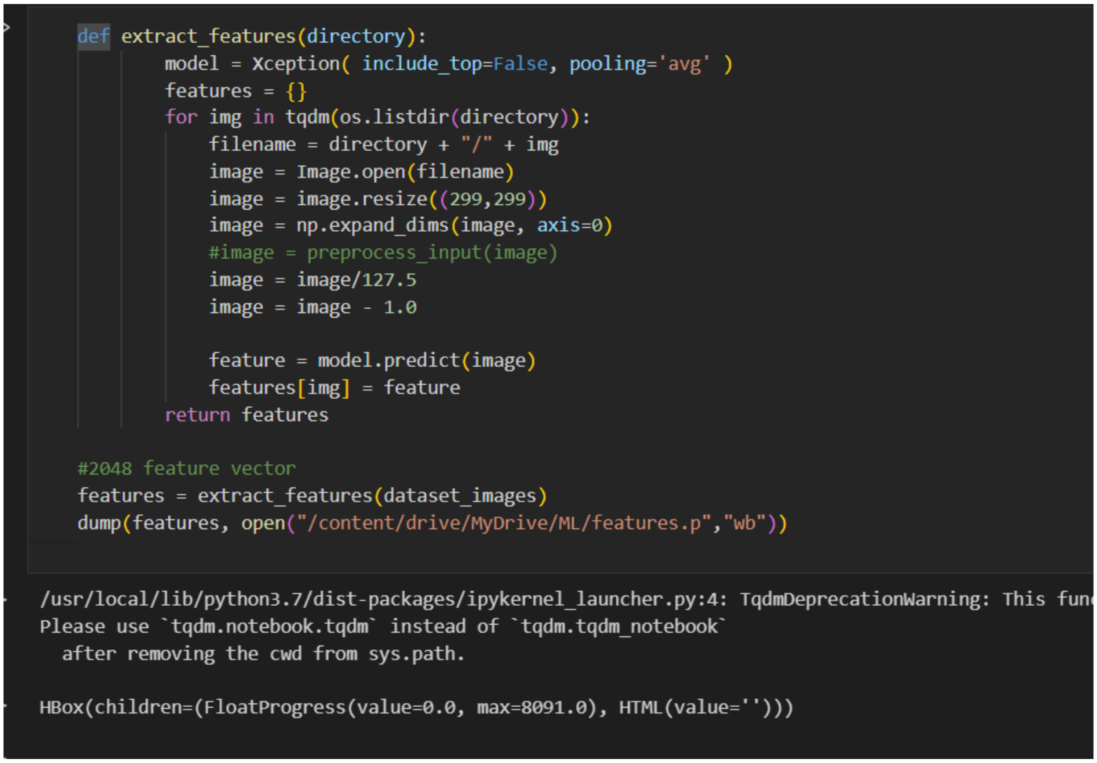


```
def text_vocabulary(descriptions):
    # build vocabulary of all unique words
    vocab = set()

    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()
```



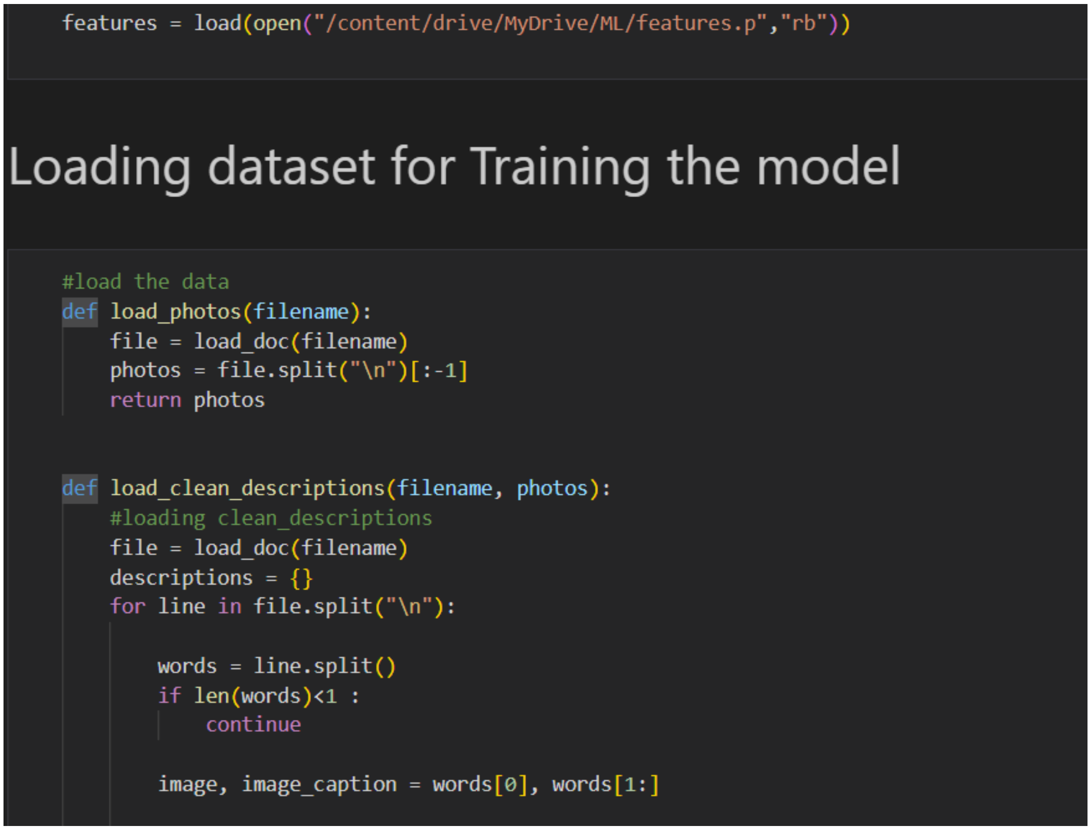
```
# Set these path according to project folder in you system
dataset_text = "/content/drive/MyDrive/ML/Flickr8k_text"
dataset_images = "/content/drive/MyDrive/ML/Flicker8k_Dataset"

#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, "/content/drive/MyDrive/ML/descriptions.txt")
```

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



```
def extract_features(directory):
        model = Xception( include_top=False, pooling='avg' )
        features = {}
        for img in tqdm(os.listdir(directory)):
            filename = directory + "/" + img
            image = Image.open(filename)
            image = image.resize((299,299))
            image = np.expand_dims(image, axis=0)
            #image = preprocess_input(image)
            image = image/127.5
            image = image - 1.0

            feature = model.predict(image)
            features[img] = feature
        return features

#2048 feature vector
features = extract_features(dataset_images)
dump(features, open("/content/drive/MyDrive/ML/features.p","wb"))
```

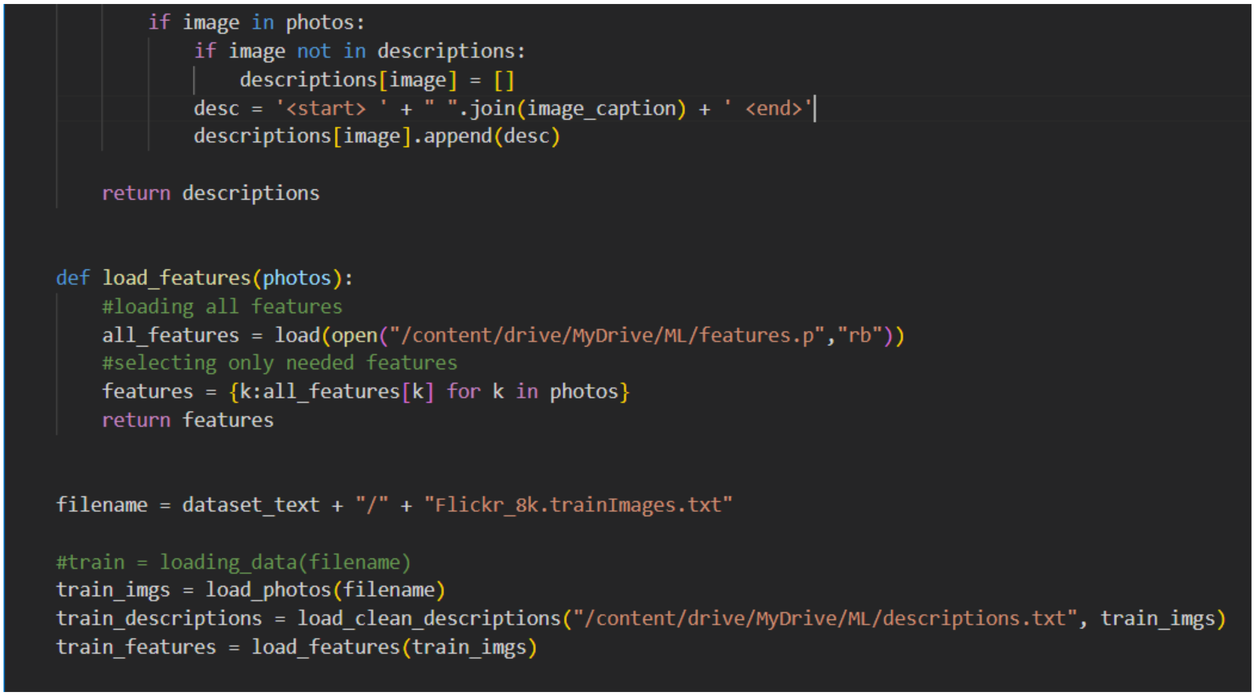
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: TqdmDeprecationWarning: This fun
Please use `tqdm.notebook.tqdm` instead of `tqdm.tqdm_notebook`
  after removing the cwd from sys.path.

HBox(children=(FloatProgress(value=0.0, max=8091.0), HTML(value='')))
```



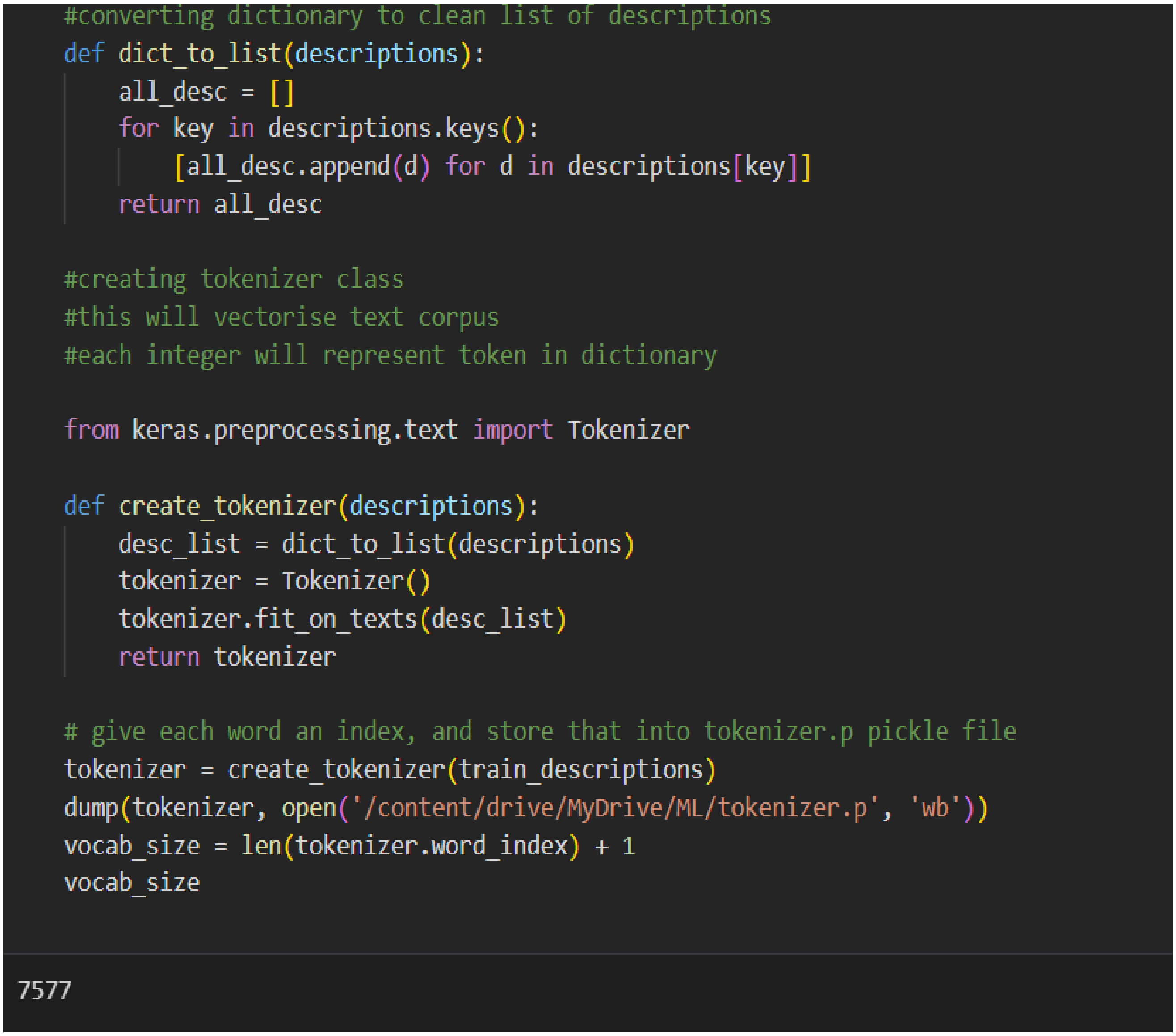
```
features = load(open("/content/drive/MyDrive/ML/features.p","rb"))
```

Loading dataset for Training the model

```
#load the data
def load_photos(filename):
    file = load_doc(filename)
    photos = file.split("\n")[:-1]
    return photos


def load_clean_descriptions(filename, photos):
    #loading clean_descriptions
    file = load_doc(filename)
    descriptions = {}
    for line in file.split("\n"):

        words = line.split()
        if len(words)<1 :
            continue

        image, image_caption = words[0], words[1:]
```

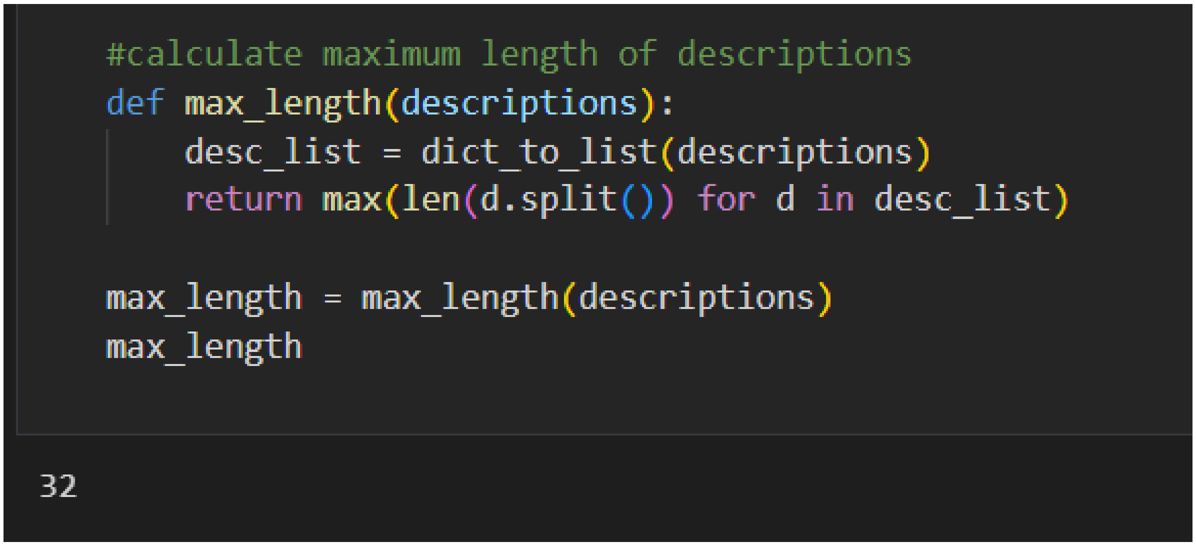


```
        if image in photos:
            if image not in descriptions:
                descriptions[image] = []
            desc = '<start> ' + " ".join(image_caption) + ' <end>'
            descriptions[image].append(desc)

    return descriptions


def load_features(photos):
    #loading all features
    all_features = load(open("/content/drive/MyDrive/ML/features.p","rb"))
    #selecting only needed features
    features = {k:all_features[k] for k in photos}
    return features


filename = dataset_text + "/" + "Flickr_8k.trainImages.txt"

#train = loading_data(filename)
train_imgs = load_photos(filename)
train_descriptions = load_clean_descriptions("/content/drive/MyDrive/ML/descriptions.txt", train_imgs)
train_features = load_features(train_imgs)
```



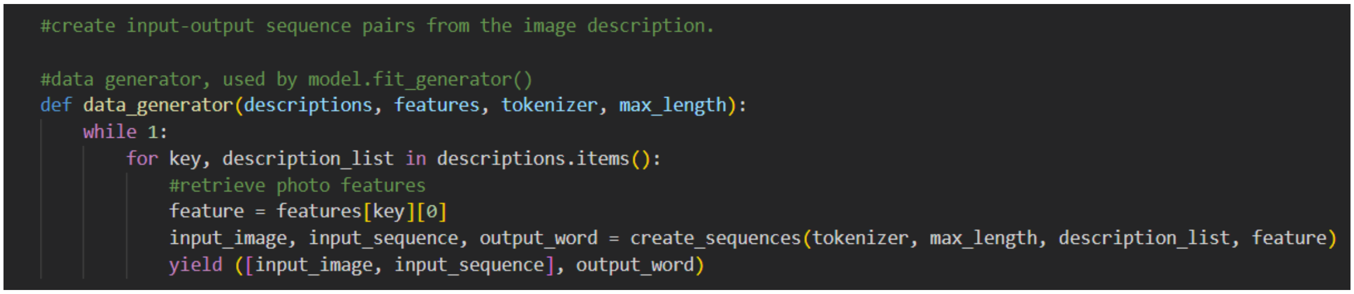
```
#converting dictionary to clean list of descriptions
def dict_to_list(descriptions):
    all_desc = []
    for key in descriptions.keys():
        [all_desc.append(d) for d in descriptions[key]]
    return all_desc

#creating tokenizer class
#this will vectorise text corpus
#each integer will represent token in dictionary

from keras.preprocessing.text import Tokenizer

def create_tokenizer(descriptions):
    desc_list = dict_to_list(descriptions)
    tokenizer = Tokenizer()
    tokenizer.fit_on_texts(desc_list)
    return tokenizer

# give each word an index, and store that into tokenizer.p pickle file
tokenizer = create_tokenizer(train_descriptions)
dump(tokenizer, open('/content/drive/MyDrive/ML/tokenizer.p', 'wb'))
vocab_size = len(tokenizer.word_index) + 1
vocab_size
```

```
7577
```

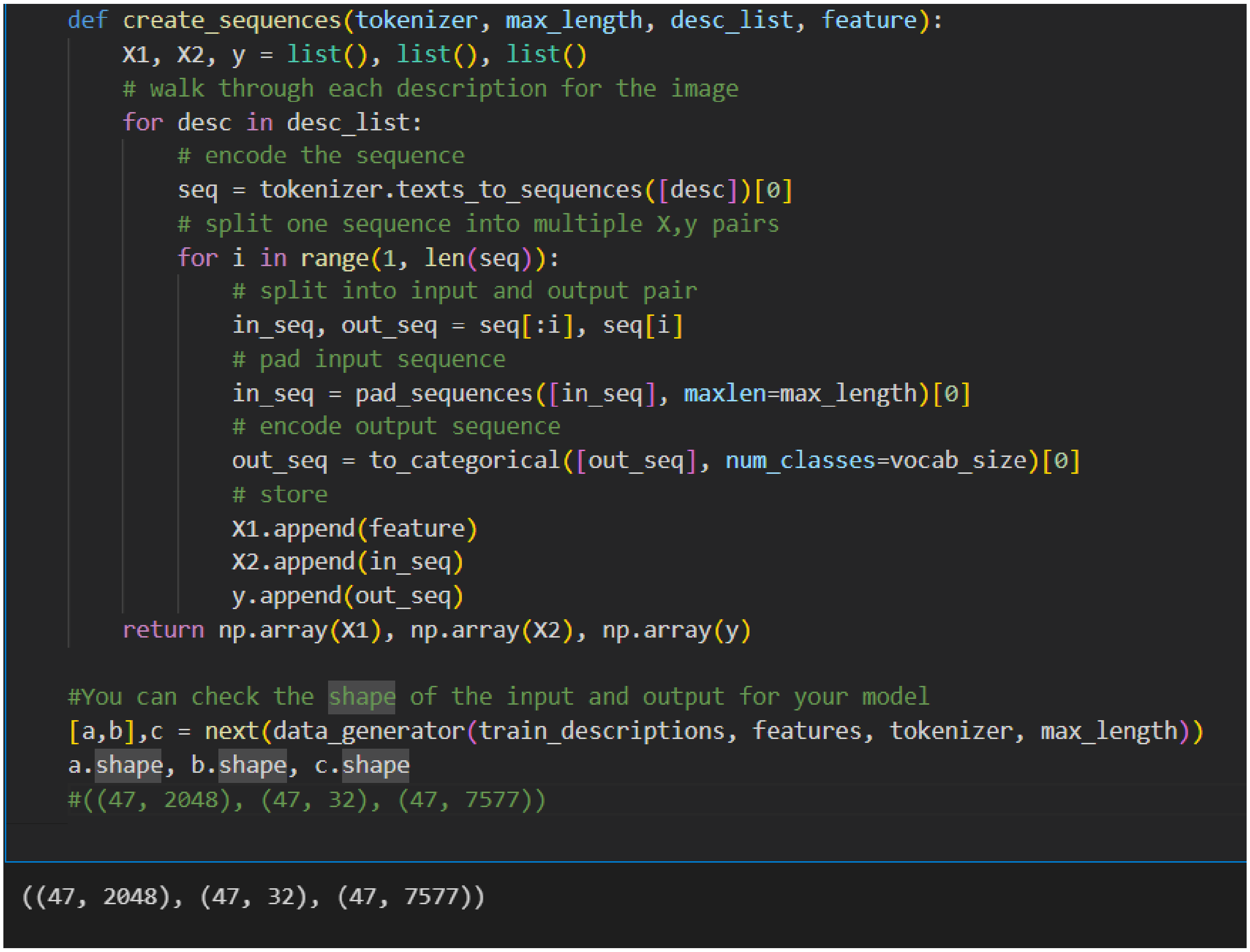


```
#calculate maximum length of descriptions
def max_length(descriptions):
    desc_list = dict_to_list(descriptions)
    return max(len(d.split()) for d in desc_list)

max_length = max_length(descriptions)
max_length
```

```
32
```

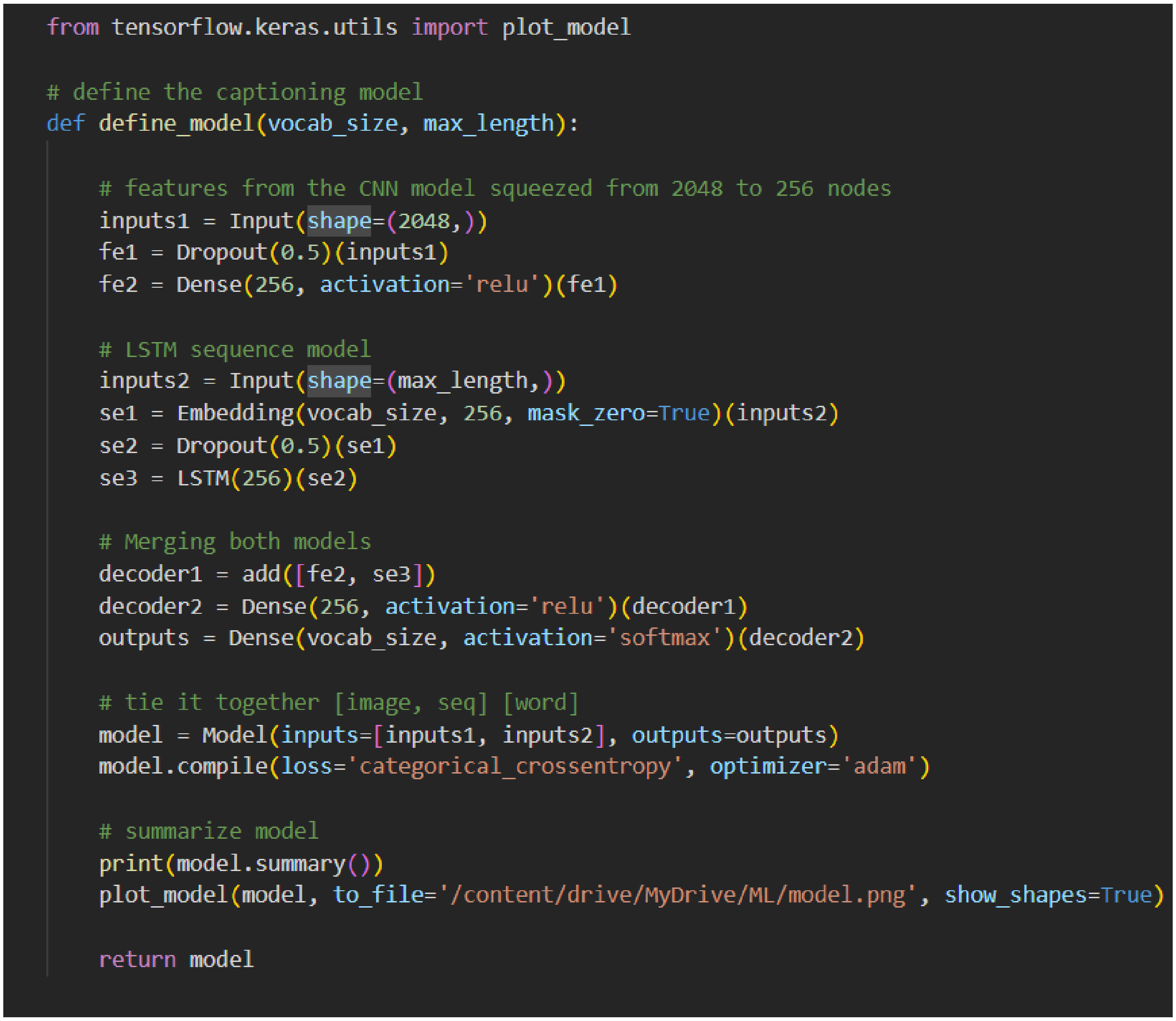


```
#create input-output sequence pairs from the image description.

#data generator, used by model.fit_generator()
def data_generator(descriptions, features, tokenizer, max_length):
    while 1:
        for key, description_list in descriptions.items():
            #retrieve photo features
            feature = features[key][0]
            input_image, input_sequence, output_word = create_sequences(tokenizer, max_length, description_list, feature)
            yield ([input_image, input_sequence], output_word)
```

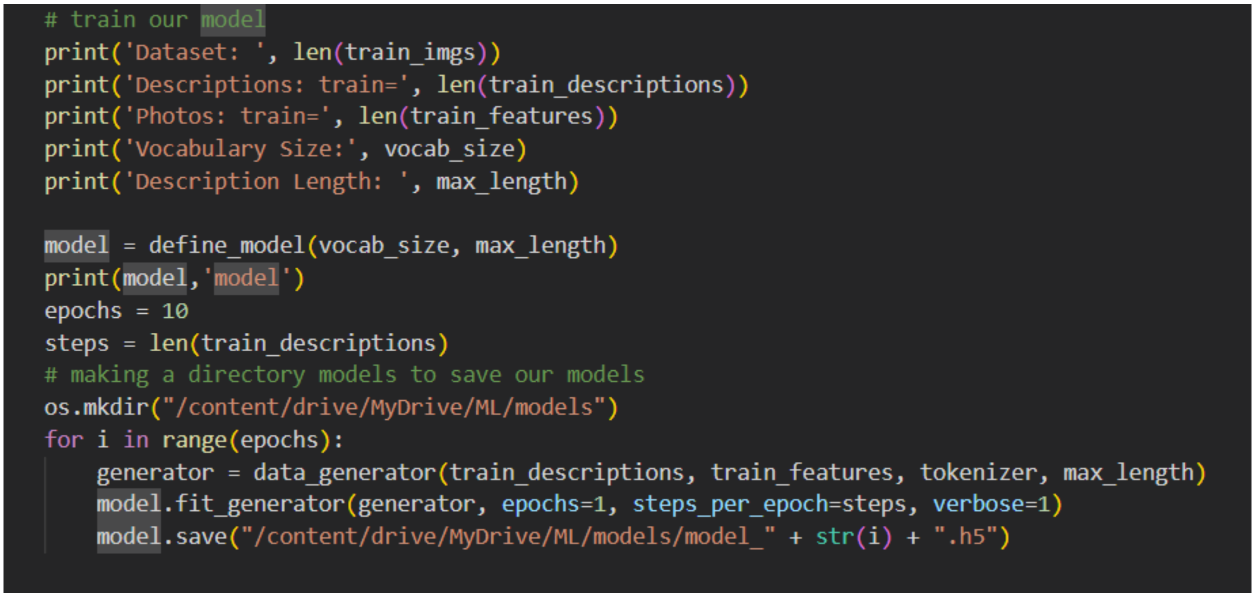


```
def create_sequences(tokenizer, max_length, desc_list, feature):
    X1, X2, y = list(), list(), list()
    # walk through each description for the image
    for desc in desc_list:
        # encode the sequence
        seq = tokenizer.texts_to_sequences([desc])[0]
        # split one sequence into multiple X,y pairs
        for i in range(1, len(seq)):
            # split into input and output pair
            in_seq, out_seq = seq[:i], seq[i]
            # pad input sequence
            in_seq = pad_sequences([in_seq], maxlen=max_length)[0]
            # encode output sequence
            out_seq = to_categorical([out_seq], num_classes=vocab_size)[0]
            # store
            X1.append(feature)
            X2.append(in_seq)
            y.append(out_seq)
    return np.array(X1), np.array(X2), np.array(y)

#You can check the shape of the input and output for your model
[a,b],c = next(data_generator(train_descriptions, features, tokenizer, max_length))
a.shape, b.shape, c.shape
#((47, 2048), (47, 32), (47, 7577))
```

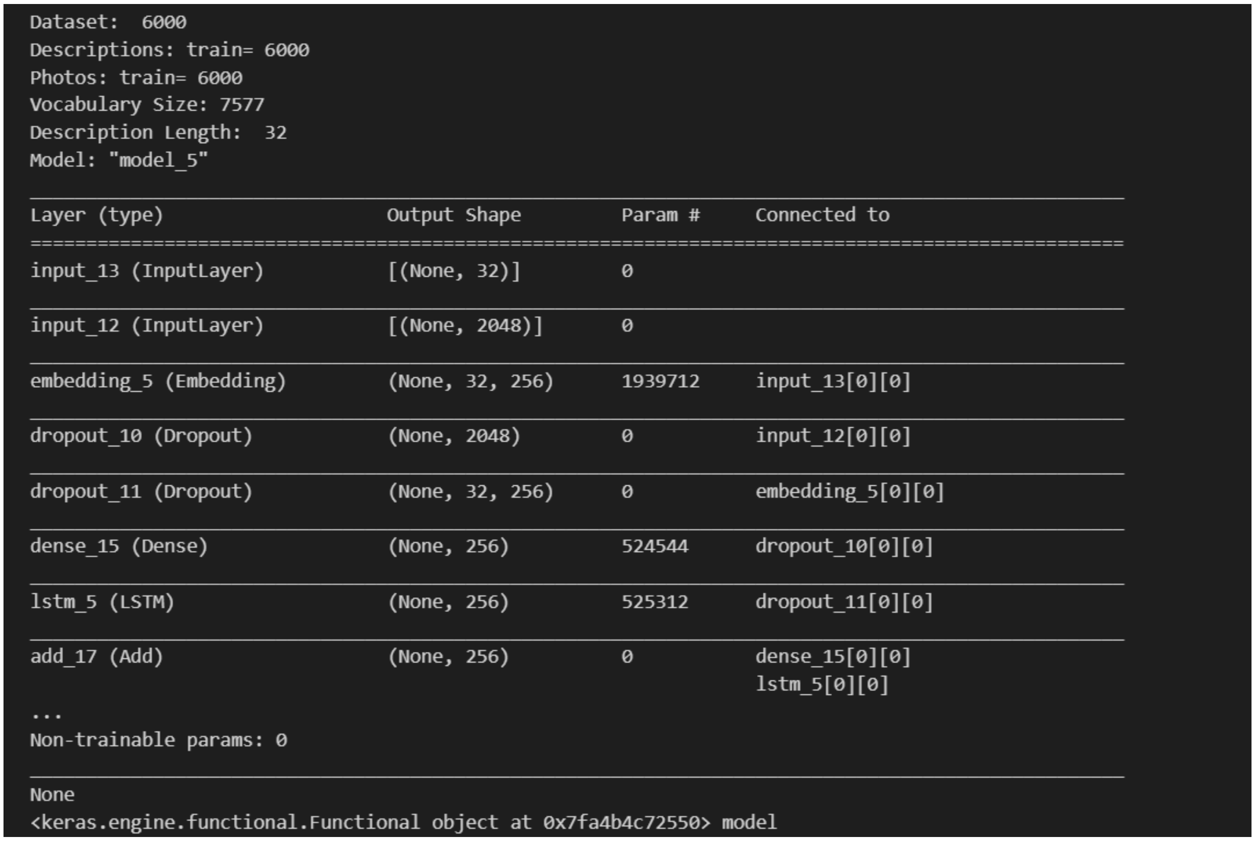
```
((47, 2048), (47, 32), (47, 7577))
```



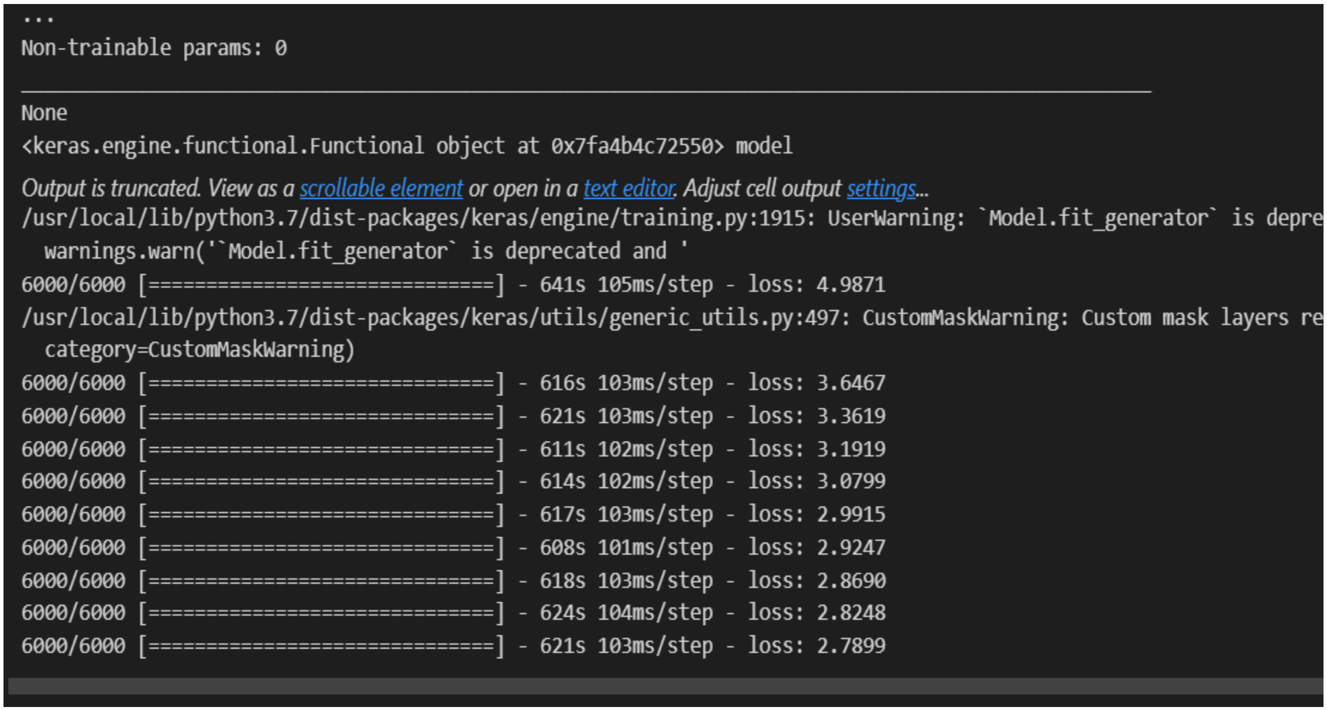
```python
from tensorflow.keras.utils import plot_model

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

    # features from the CNN model squeezed from 2048 to 256 nodes
    inputs1 = Input(shape=(2048,))
    fe1 = Dropout(0.5)(inputs1)
    fe2 = Dense(256, activation='relu')(fe1)

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

    # Merging both models
    decoder1 = add([fe2, se3])
    decoder2 = Dense(256, activation='relu')(decoder1)
    outputs = Dense(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())
    plot_model(model, to_file='/content/drive/MyDrive/ML/model.png', show_shapes=True)

    return model
```



```python
# train our model
print('Dataset: ', len(train_imgs))
print('Descriptions: train=', len(train_descriptions))
print('Photos: train=', len(train_features))
print('Vocabulary Size:', vocab_size)
print('Description Length: ', max_length)

model = define_model(vocab_size, max_length)
print(model,'model')
epochs = 10
steps = len(train_descriptions)
# making a directory models to save our models
os.mkdir("/content/drive/MyDrive/ML/models")
for i in range(epochs):
    generator = data_generator(train_descriptions, train_features, tokenizer, max_length)
    model.fit_generator(generator, epochs=1, steps_per_epoch=steps, verbose=1)
    model.save("/content/drive/MyDrive/ML/models/model_" + str(i) + ".h5")
```



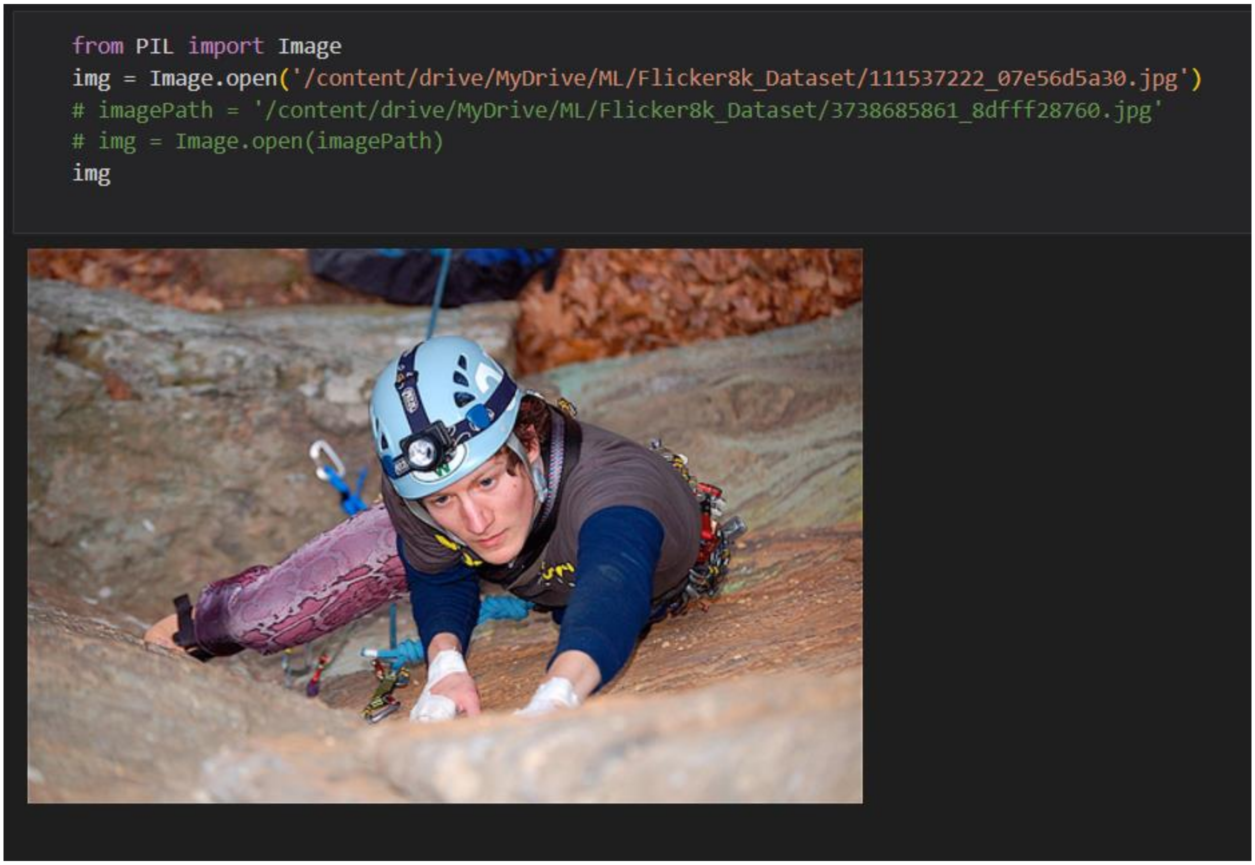
```
Dataset:  6000
Descriptions: train= 6000
Photos: train= 6000
Vocabulary Size: 7577
Description Length:  32
Model: "model_5"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_13 (InputLayer)           [(None, 32)]         0
__________________________________________________________________________________________________
input_12 (InputLayer)           [(None, 2048)]       0
__________________________________________________________________________________________________
embedding_5 (Embedding)         (None, 32, 256)      1939712     input_13[0][0]
__________________________________________________________________________________________________
dropout_10 (Dropout)            (None, 2048)         0           input_12[0][0]
__________________________________________________________________________________________________
dropout_11 (Dropout)            (None, 32, 256)      0           embedding_5[0][0]
__________________________________________________________________________________________________
dense_15 (Dense)                (None, 256)          524544      dropout_10[0][0]
__________________________________________________________________________________________________
lstm_5 (LSTM)                   (None, 256)          525312      dropout_11[0][0]
__________________________________________________________________________________________________
add_17 (Add)                    (None, 256)          0           dense_15[0][0]
                                                                 lstm_5[0][0]
...
Non-trainable params: 0
__________________________________________________________________________________________________
None
<keras.engine.functional.Functional object at 0x7fa4b4c72550> model
```



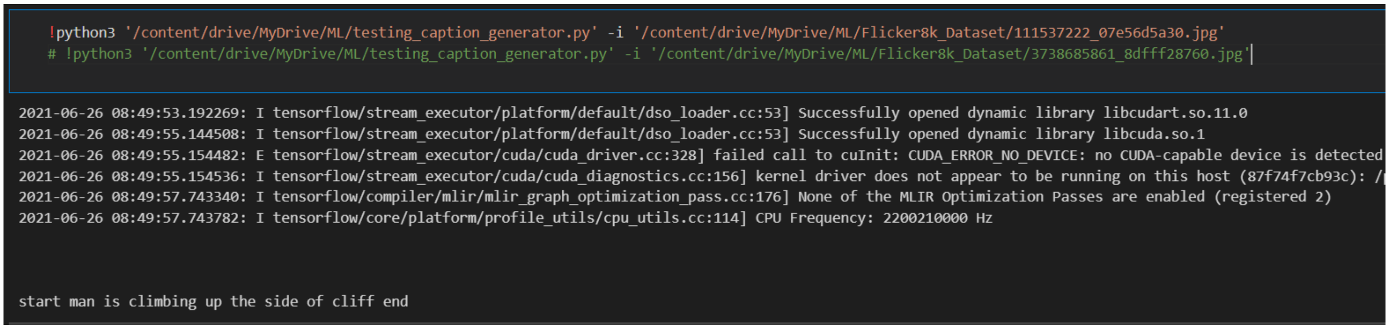
```
...
Non-trainable params: 0
__________________________________________________________________________________________________
None
<keras.engine.functional.Functional object at 0x7fa4b4c72550> model
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```
/usr/local/lib/python3.7/dist-packages/keras/engine/training.py:1915: UserWarning: `Model.fit_generator` is depre
  warnings.warn('`Model.fit_generator` is deprecated and '
6000/6000 [==============================] - 641s 105ms/step - loss: 4.9871
/usr/local/lib/python3.7/dist-packages/keras/utils/generic_utils.py:497: CustomMaskWarning: Custom mask layers re
  category=CustomMaskWarning)
6000/6000 [==============================] - 616s 103ms/step - loss: 3.6467
6000/6000 [==============================] - 621s 103ms/step - loss: 3.3619
6000/6000 [==============================] - 611s 102ms/step - loss: 3.1919
6000/6000 [==============================] - 614s 102ms/step - loss: 3.0799
6000/6000 [==============================] - 617s 103ms/step - loss: 2.9915
6000/6000 [==============================] - 608s 101ms/step - loss: 2.9247
6000/6000 [==============================] - 618s 103ms/step - loss: 2.8690
6000/6000 [==============================] - 624s 104ms/step - loss: 2.8248
6000/6000 [==============================] - 621s 103ms/step - loss: 2.7899
```

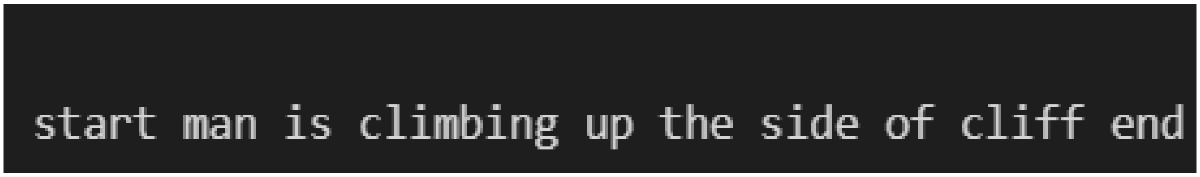


```
from PIL import Image
img = Image.open('/content/drive/MyDrive/ML/Flicker8k_Dataset/111537222_07e56d5a30.jpg')
# imagePath = '/content/drive/MyDrive/ML/Flicker8k_Dataset/3738685861_8dfff28760.jpg'
# img = Image.open(imagePath)
img
```



```
!python3 '/content/drive/MyDrive/ML/testing_caption_generator.py' -i '/content/drive/MyDrive/ML/Flicker8k_Dataset/111537222_07e56d5a30.jpg'
# !python3 '/content/drive/MyDrive/ML/testing_caption_generator.py' -i '/content/drive/MyDrive/ML/Flicker8k_Dataset/3738685861_8dfff28760.jpg'
```

```
2021-06-26 08:49:53.192269: I tensorflow/stream_executor/platform/default/dso_loader.cc:53] Successfully opened dynamic library libcudart.so.11.0
2021-06-26 08:49:55.144508: I tensorflow/stream_executor/platform/default/dso_loader.cc:53] Successfully opened dynamic library libcuda.so.1
2021-06-26 08:49:55.154482: E tensorflow/stream_executor/cuda/cuda_driver.cc:328] failed call to cuInit: CUDA_ERROR_NO_DEVICE: no CUDA-capable device is detected
2021-06-26 08:49:55.154536: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:156] kernel driver does not appear to be running on this host (87f74f7cb93c): /
2021-06-26 08:49:57.743340: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:176] None of the MLIR Optimization Passes are enabled (registered 2)
2021-06-26 08:49:57.743782: I tensorflow/core/platform/profile_utils/cpu_utils.cc:114] CPU Frequency: 2200210000 Hz


start man is climbing up the side of cliff end
```



```
start man is climbing up the side of cliff end
```