Data Collection and Preprocessing Phase

Date	7 November 2024
Team ID	739889
Project Title	Image Caption Generator
Maximum Marks	6 Marks

Preprocessing Template

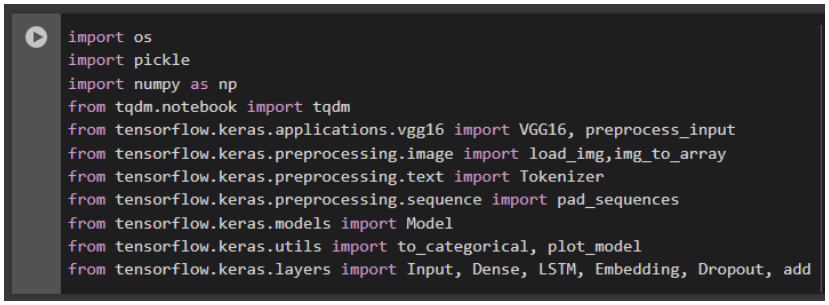
The images will be preprocessed by resizing, normalizing, augmenting, denoising, adjusting contrast, detecting edges, converting color space, cropping, batch normalizing, and whitening data. These steps will enhance data quality, promote model generalization, and improve convergence during neural network training, ensuring robust and efficient performance across various computer vision tasks.

Section	**Description**
Data Overview	This stage would involve providing a description of the image dataset that the captioning model will be trained and evaluated on.
Resizing	Images in the dataset would be resized to a uniform target size to ensure consistent input for the model.
Normalization	Pixel values of the images would be normalized to a specific range, which can help in stabilizing the training process and improving model performance.
Data Augmentation	Techniques such as flipping, rotation, shifting, zooming, or shearing would be applied to artificially increase the size and diversity of the training data.
Denoising	Filters might be applied to reduce noise within the images, potentially leading to cleaner feature extraction.
Edge Detection	Algorithms would be used to identify and highlight significant edges in the images, which can provide important structural information.

Color Space Conversion	Images could be converted from one color space to another, possibly to emphasize certain color characteristics or simplify the input.
Image Cropping	Crop images to focus on the regions containing objects of interest.
Batch Normalization	Apply batch normalization to the input of each layer in the neural network.

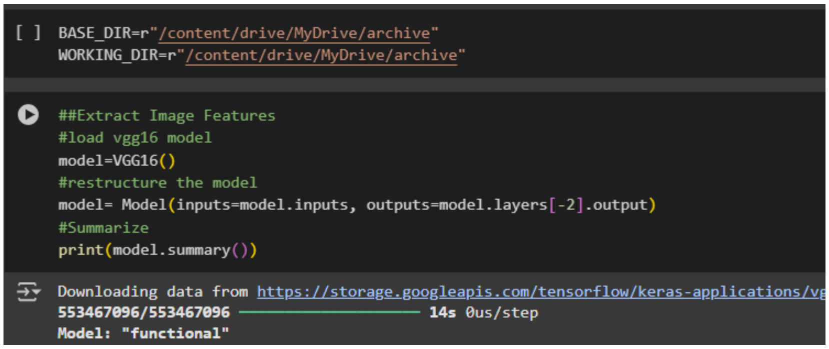
Data Preprocessing Code Screenshots

Loading Data



```
import os
import pickle
import numpy as np
from tqdm.notebook import tqdm
from tensorflow.keras.applications.vgg16 import VGG16, preprocess_input
from tensorflow.keras.preprocessing.image import load_img,img_to_array
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Model
from tensorflow.keras.utils import to_categorical, plot_model
from tensorflow.keras.layers import Input, Dense, LSTM, Embedding, Dropout, add
```

Extracting Images

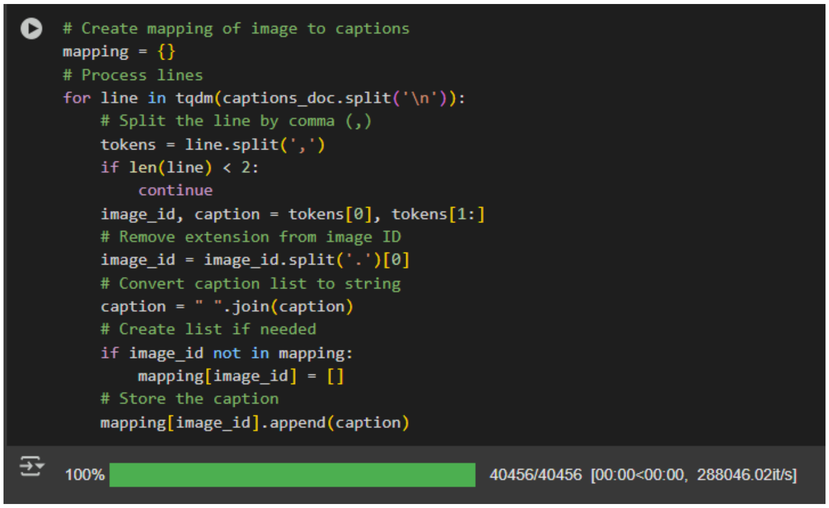


```
[ ] BASE_DIR=r"/content/drive/MyDrive/archive"
    WORKING_DIR=r"/content/drive/MyDrive/archive"
```

```
##Extract Image Features
#load vgg16 model
model=VGG16()
#restructure the model
model= Model(inputs=model.inputs, outputs=model.layers[-2].output)
#Summarize
print(model.summary())
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vg
553467096/553467096 ━━━━━━━━━━━━━━━━━━━━ 14s 0us/step
Model: "functional"
```

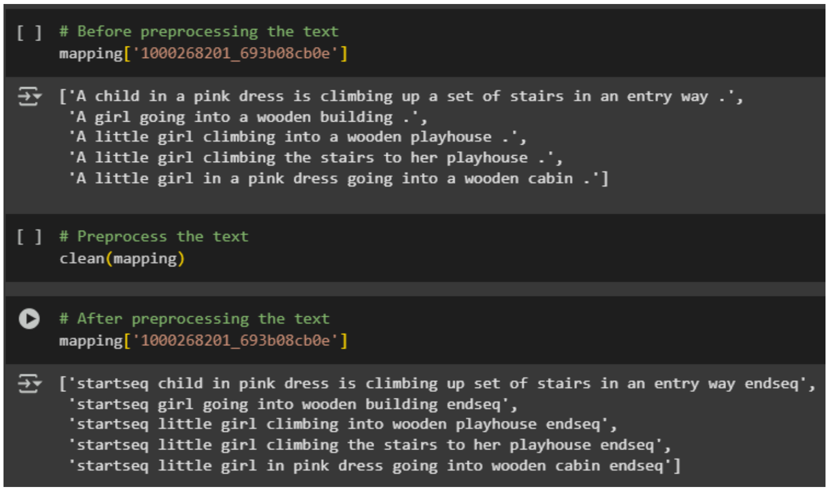
Mapping Image to Captiona



```
# Create mapping of image to captions
mapping = {}
# Process lines
for line in tqdm(captions_doc.split('\n')):
    # Split the line by comma (,)
    tokens = line.split(',')
    if len(line) < 2:
        continue
    image_id, caption = tokens[0], tokens[1:]
    # Remove extension from image ID
    image_id = image_id.split('.')[0]
    # Convert caption list to string
    caption = " ".join(caption)
    # Create list if needed
    if image_id not in mapping:
        mapping[image_id] = []
    # Store the caption
    mapping[image_id].append(caption)
```

```
100% ████████████████████ 40456/40456 [00:00<00:00, 288046.02it/s]
```

<table>
<tr>
<td>Preprocessing the Text</td>
<td>



```
[ ] # Before preprocessing the text
    mapping['1000268201_693b08cb0e']
```

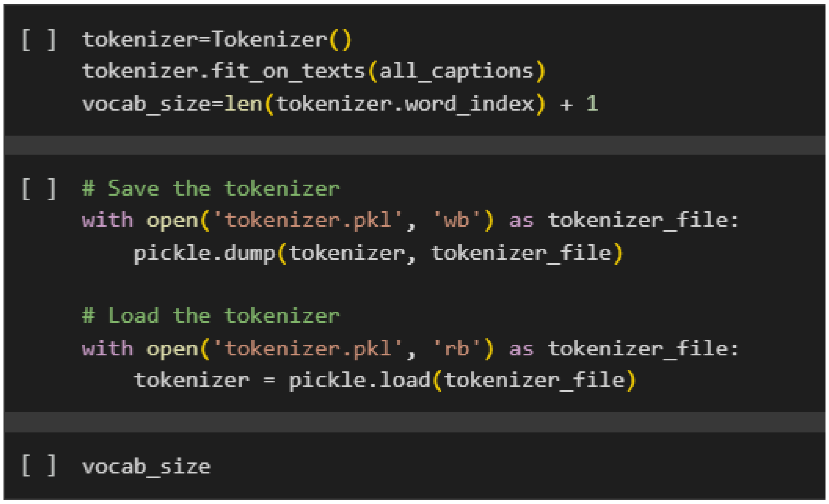
```
['A child in a pink dress is climbing up a set of stairs in an entry way .',
 'A girl going into a wooden building .',
 'A little girl climbing into a wooden playhouse .',
 'A little girl climbing the stairs to her playhouse .',
 'A little girl in a pink dress going into a wooden cabin .']
```

```
[ ] # Preprocess the text
    clean(mapping)
```

```
# After preprocessing the text
mapping['1000268201_693b08cb0e']
```

```
['startseq child in pink dress is climbing up set of stairs in an entry way endseq',
 'startseq girl going into wooden building endseq',
 'startseq little girl climbing into wooden playhouse endseq',
 'startseq little girl climbing the stairs to her playhouse endseq',
 'startseq little girl in pink dress going into wooden cabin endseq']
```

</td>
</tr>
<tr>
<td>Tokenizer</td>
<td>

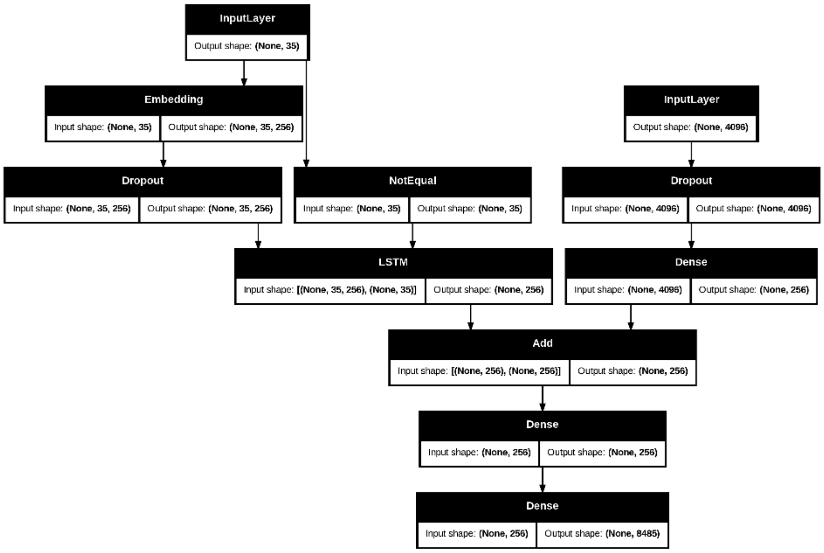


```
[ ] tokenizer=Tokenizer()
    tokenizer.fit_on_texts(all_captions)
    vocab_size=len(tokenizer.word_index) + 1
```

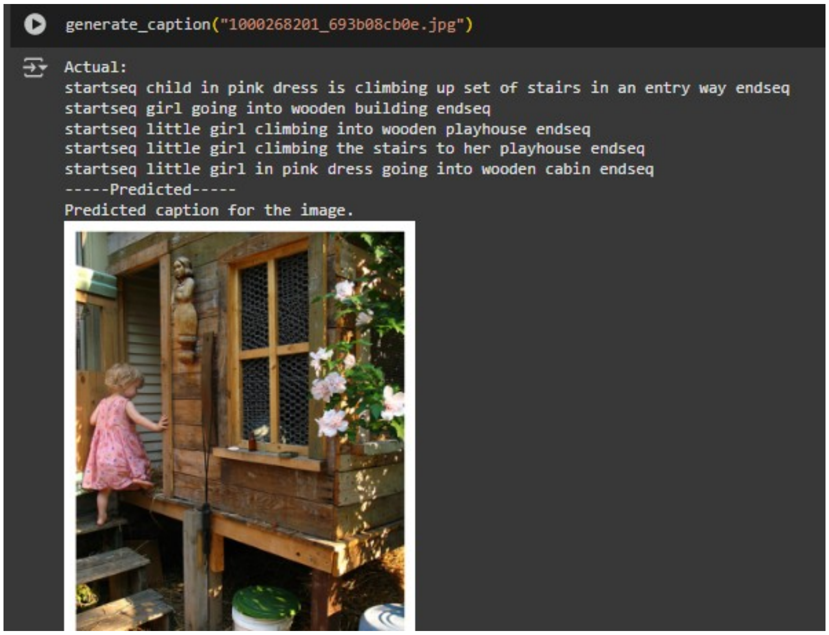
```
[ ] # Save the tokenizer
    with open('tokenizer.pkl', 'wb') as tokenizer_file:
        pickle.dump(tokenizer, tokenizer_file)

    # Load the tokenizer
    with open('tokenizer.pkl', 'rb') as tokenizer_file:
        tokenizer = pickle.load(tokenizer_file)
```

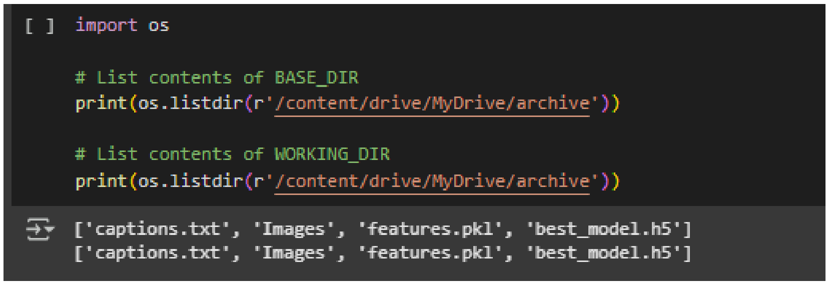
```
[ ] vocab_size
```

</td>
</tr>
<tr>
<td>Model Architecture</td>
<td>


InputLayer
Output shape: (None, 35)
Embedding
Input shape: (None, 35)
Output shape: (None, 35, 256)
InputLayer
Output shape: (None, 4096)
Dropout
Input shape: (None, 35, 256)
Output shape: (None, 35, 256)
NotEqual
Input shape: (None, 35)
Output shape: (None, 35)
Dropout
Input shape: (None, 4096)
Output shape: (None, 4096)
LSTM
Input shape: [(None, 35, 256), (None, 35)]
Output shape: (None, 256)
Dense
Input shape: (None, 4096)
Output shape: (None, 256)
Add
Input shape: [(None, 256), (None, 256)]
Output shape: (None, 256)
Dense
Input shape: (None, 256)
Output shape: (None, 256)
Dense
Input shape: (None, 256)
Output shape: (None, 8485)

</td>
</tr>
</table>

Generate Cption	 generate_caption("1000268201_693b08cb0e.jpg") Actual: startseq child in pink dress is climbing up set of stairs in an entry way endseq startseq girl going into wooden building endseq startseq little girl climbing into wooden playhouse endseq startseq little girl climbing the stairs to her playhouse endseq startseq little girl in pink dress going into wooden cabin endseq -----Predicted----- Predicted caption for the image.

Extracting the model	
	import os
	
	# List contents of BASE_DIR
	print(os.listdir(r'/content/drive/MyDrive/archive'))
	
	# List contents of WORKING_DIR
	print(os.listdir(r'/content/drive/MyDrive/archive'))
	
	['captions.txt', 'Images', 'features.pkl', 'best_model.h5']
	['captions.txt', 'Images', 'features.pkl', 'best_model.h5']