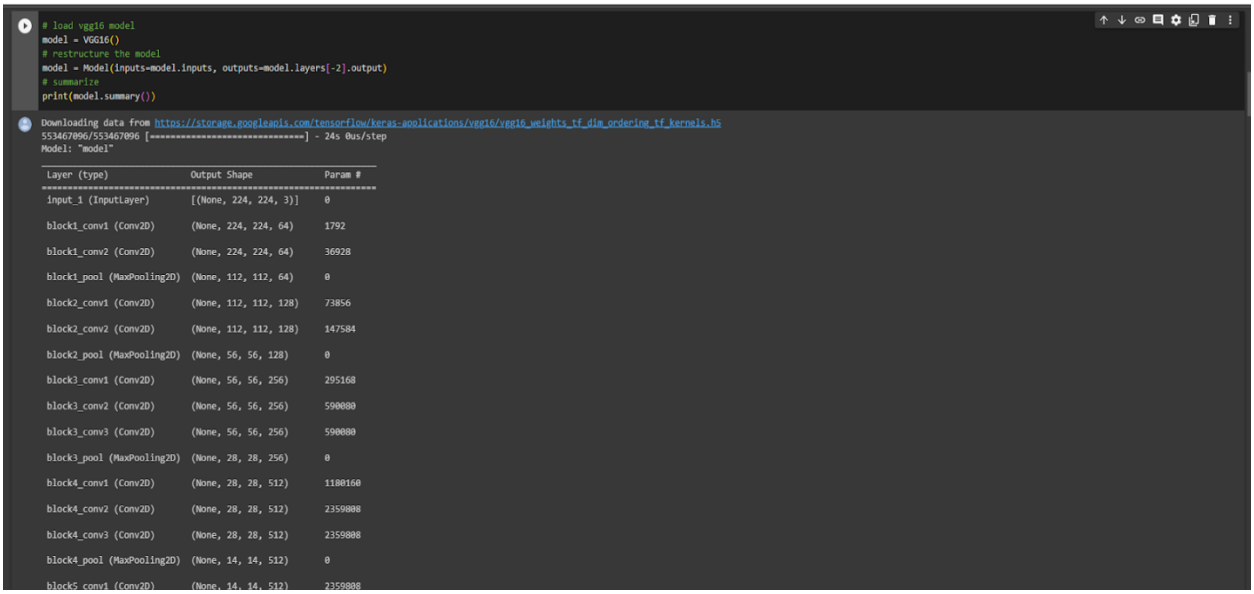
Model Development Phase Template

Date	12 November 2024
Team ID	team-739761
Project Title	PixelProse - Crafting Visual Stories with Intelligent Image Captioning
Maximum Marks	10 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include a summary and training and validation performance metrics for multiple models, presented through respective screenshots.

Initial Model Training Code (5 marks):



```
# 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/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels.h5
553467096/553467096 [==============================] - 24s 0us/step
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0

 block4_conv1 (Conv2D)       (None, 28, 28, 512)       1180160

 block4_conv2 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv3 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 14, 14, 512)       0

 block5_conv1 (Conv2D)       (None, 14, 14, 512)       2359808
```

```
model=get_model()
model.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet152v2_weights_tf_dim_ordering_tf_kernels_notop.h5
234545216/234545216 ━━━━━━━━━━━━━━━━━━━━ 1s 0us/step
Model: "functional"
```

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 256, 256, 3)	0	-
conv1_pad (ZeroPadding2D)	(None, 262, 262, 3)	0	input_layer[0][0]
conv1_conv (Conv2D)	(None, 128, 128, 64)	9,472	conv1_pad[0][0]
pool1_pad (ZeroPadding2D)	(None, 130, 130, 64)	0	conv1_conv[0][0]
pool1_pool (MaxPooling2D)	(None, 64, 64, 64)	0	pool1_pad[0][0]
conv2_block1_preact_bn (BatchNormalization)	(None, 64, 64, 64)	256	pool1_pool[0][0]
conv2_block1_preact_relu (Activation)	(None, 64, 64, 64)	0	conv2_block1_preact_b…
conv2_block1_1_conv (Conv2D)	(None, 64, 64, 64)	4,096	conv2_block1_preact_r…
conv2_block1_1_bn (BatchNormalization)	(None, 64, 64, 64)	256	conv2_block1_1_conv[0…
conv2_block1_1_relu (Activation)	(None, 64, 64, 64)	0	conv2_block1_1_bn[0][…
conv2_block1_2_pad (ZeroPadding2D)	(None, 66, 66, 64)	0	conv2_block1_1_relu[0…

conv5_block3_2_pad (ZeroPadding2D)	(None, 10, 10, 512)	0	conv5_block3_1_relu[0…
conv5_block3_2_conv (Conv2D)	(None, 8, 8, 512)	2,359,296	conv5_block3_2_pad[0]…
conv5_block3_2_bn (BatchNormalization)	(None, 8, 8, 512)	2,048	conv5_block3_2_conv[0…
conv5_block3_2_relu (Activation)	(None, 8, 8, 512)	0	conv5_block3_2_bn[0][…
conv5_block3_3_conv (Conv2D)	(None, 8, 8, 2048)	1,050,624	conv5_block3_2_relu[0…
conv5_block3_out (Add)	(None, 8, 8, 2048)	0	conv5_block2_out[0][0… conv5_block3_3_conv[0…
post_bn (BatchNormalization)	(None, 8, 8, 2048)	8,192	conv5_block3_out[0][0]
post_relu (Activation)	(None, 8, 8, 2048)	0	post_bn[0][0]
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0	post_relu[0][0]
dense (Dense)	(None, 1000)	2,049,000	global_average_poolin…
dense_1 (Dense)	(None, 10)	10,010	dense[0][0]

```
Total params: 60,390,658 (230.37 MB)
Trainable params: 56,430,338 (215.26 MB)
Non-trainable params: 3,960,320 (15.11 MB)
```

+ Code + Text Connect

```python
# train the model
epochs = 20
batch_size = 32
steps = len(train) // batch_size

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

```
227/227 [==============================] - 58s 254ms/step - loss: 3.9065
227/227 [==============================] - 57s 251ms/step - loss: 3.5282
227/227 [==============================] - 58s 255ms/step - loss: 3.2727
227/227 [==============================] - 57s 250ms/step - loss: 3.0780
227/227 [==============================] - 57s 253ms/step - loss: 2.9318
227/227 [==============================] - 56s 248ms/step - loss: 2.8202
227/227 [==============================] - 58s 253ms/step - loss: 2.7285
227/227 [==============================] - 56s 246ms/step - loss: 2.6544
227/227 [==============================] - 57s 249ms/step - loss: 2.5852
227/227 [==============================] - 56s 244ms/step - loss: 2.5283
227/227 [==============================] - 57s 251ms/step - loss: 2.4743
227/227 [==============================] - 56s 247ms/step - loss: 2.4271
227/227 [==============================] - 57s 252ms/step - loss: 2.3788
227/227 [==============================] - 57s 249ms/step - loss: 2.3370
227/227 [==============================] - 57s 251ms/step - loss: 2.2964
227/227 [==============================] - 57s 249ms/step - loss: 2.2610
227/227 [==============================] - 58s 255ms/step - loss: 2.2289
227/227 [==============================] - 57s 250ms/step - loss: 2.1955
227/227 [==============================] - 57s 253ms/step - loss: 2.1702
227/227 [==============================] - 56s 247ms/step - loss: 2.1434
```

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

```python
len(all_captions)
```

```
40455
```

```python
all_captions[:10]
```

```
['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',
 'startseq black dog and spotted dog are fighting endseq',
 'startseq black dog and tri-colored dog playing with each other on the road endseq',
 'startseq black dog and white dog with brown spots are staring at each other in the street endseq',
 'startseq two dogs of different breeds looking at each other on the road endseq',
 'startseq two dogs on pavement moving toward each other endseq']
```

```python
# tokenize the text
tokenizer = Tokenizer()
tokenizer.fit_on_texts(all_captions)
vocab_size = len(tokenizer.word_index) + 1
```

```python
vocab_size
```

```
8485
```

```python
# get maximum length of the caption available
max_length = max(len(caption.split()) for caption in all_captions)
max_length
```

```
35
```

```python
def clean(mapping):
    for key, captions in mapping.items():
        for i in range(len(captions)):
            # take one caption at a time
            caption = captions[i]
            # preprocessing steps
            # convert to lowercase
            caption = caption.lower()
            # delete digits, special chars, etc.,
            caption = caption.replace('[^A-Za-z]', '')
            # delete additional spaces
            caption = caption.replace('\s+', ' ')
            # add start and end tags to the caption
            caption = 'startseq ' + " ".join([word for word in caption.split() if len(word)>1]) + ' endseq'
            captions[i] = caption
```

```python
# before preprocess of 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 .']
```

```python
# preprocess the text
clean(mapping)
```

```python
# after preprocess of 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']
```

Model Validation and Evaluation Report (5 marks):

Model	Summary	Training and Validation Performance Metrics
InceptionV3	Inception v3 is a highly efficient and accurate convolutional neural network (CNN) architecture, widely used for image classification and feature extraction tasks. Developed as part of the Google Inception series, it improves computational efficiency while maintaining high performance. **Factorized Convolutions:** Breaks down large convolutions into smaller, more efficient ones, reducing computation time and improving performance. **Auxiliary Classifiers:** Incorporates auxiliary outputs during training to improve gradient flow and prevent overfitting. **Batch Normalization:** Normalizes input data across layers to accelerate convergence and stabilize training.