Project Development Phase
Model Performance Test

Date	15 November 2023
Team ID	598189
Project Name	Project - Image Caption Generation
Maximum Marks	10 Marks

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE - , MSE - , RMSE - , R2 score - **Classification Model:** Confusion Matrix - , Accuray Score- & Classification Report -	Blue Score - 0.14 Accuracy - 89.6
2.	Tune the Model	Hyperparameter Tuning - Validation Method -	Epochs - 200 Beam Search

```python
from nltk.translate.bleu_score import corpus_bleu
from tqdm import tqdm
import os

def generate_caption(model, image_feature, max_length=25):
    pred_text = ['startofseq']
    count = 0
    caption = ''

    while count < max_length:
        count += 1
        encoded = []

        for word in pred_text:
            encoded.append(count_words[word])

        encoded = [encoded]
        encoded = pad_sequences(encoded, maxlen=MAX_LEN, padding='post', truncating='post')
        pred_idx = np.argmax(model.predict([image_feature, encoded]))
        sampled_word = inverse_dict[pred_idx]

        if sampled_word == 'endofseq':
            break
        caption = caption + ' ' + sampled_word
        pred_text.append(sampled_word)

    return caption.strip()

predicted_captions = {}
actual_captions = {}

num_samples = 100  # You can adjust the number of samples for evaluation

for i in tqdm(range(num_samples)):
    random_no = np.random.randint(0, 1501, (1, 1))[0, 0]
    test_feature = model.predict(getImage(random_no)).reshape(1, 2048)
    test_img_path = images[random_no]
    actual_caption = captions_dict[os.path.basename(test_img_path)][0]  # Ensure test_img_path is a string

    # Convert elements to strings and join
    actual_caption = ' '.join(map(str, actual_caption))

    predicted_caption = generate_caption(final_model, test_feature)

    actual_captions[test_img_path] = [actual_caption.split()]
    predicted_captions[test_img_path] = predicted_caption.split()

# Calculate BLEU score
bleu_score = corpus_bleu(list(actual_captions.values()), list(predicted_captions.values()))

print(f"BLEU Score: {bleu_score}")
```

```
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 28ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 35ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 23ms/step
100%|██████████| 100/100 [01:31<00:00,  1.10it/s]
BLEU Score: 0.13872757777325356
```