Infosys SpringBoard Internship 4.0 (June – July 2024)

Image Captioning
MEDICAL IMAGES (X-RAYS)

Intern Name : Boddupally Sanjana
Mentor : Sudheer Kumar Y

PROBLEM STATEMENT

Develop a deep learning model to generate accurate and meaningful captions for X-ray images, automating the interpretation process to enhance efficiency and consistency in medical image analysis.

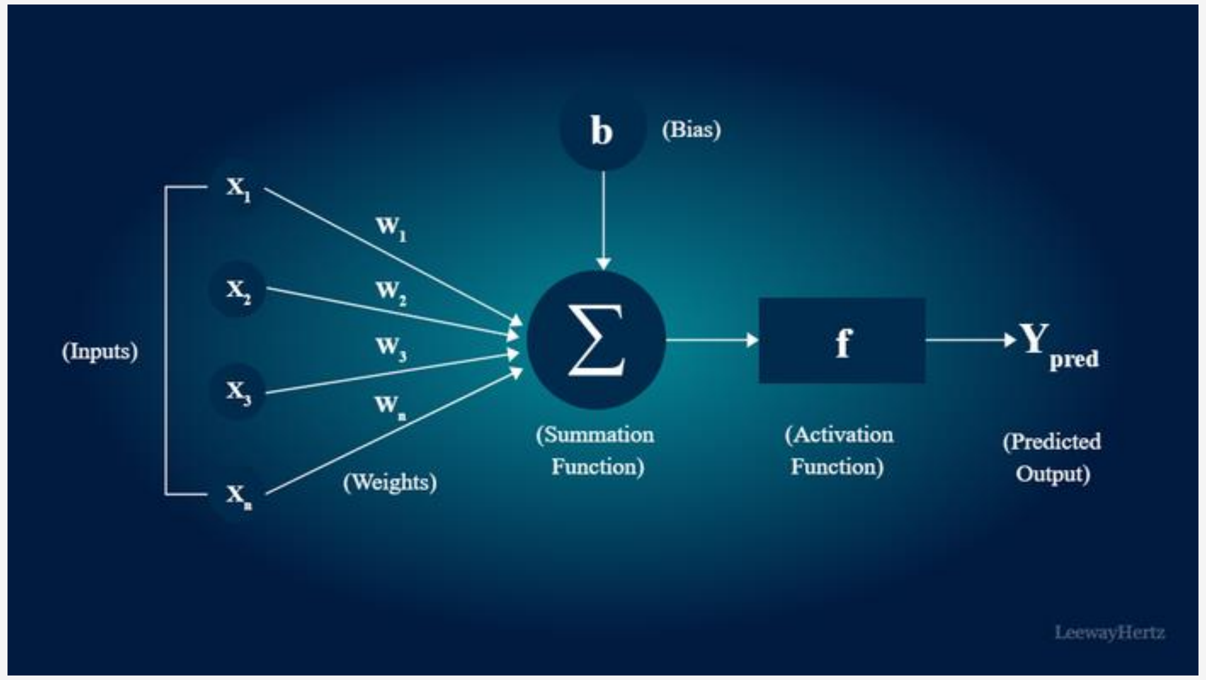


IMAGE CAPTIONING

Image Captioning is the process of automatically generating a textual description for an image using machine learning and computer vision techniques. This task involves understanding the content of the image, identifying key objects, actions, and scenes, and then expressing this understanding in natural language. Image captioning combines image processing and natural language processing, typically utilizing convolutional neural networks (CNNs) for extracting image features and recurrent neural networks (RNNs), like LSTMs, for generating coherent and contextually relevant sentences. The goal is to produce accurate, meaningful, and contextually appropriate descriptions that facilitate better understanding and communication of visual content.

USE CASES

Medical Education

Helps medical students and trainees learn by providing detailed, consistent descriptions of images for study and reference.

Automated CCTV Monitoring

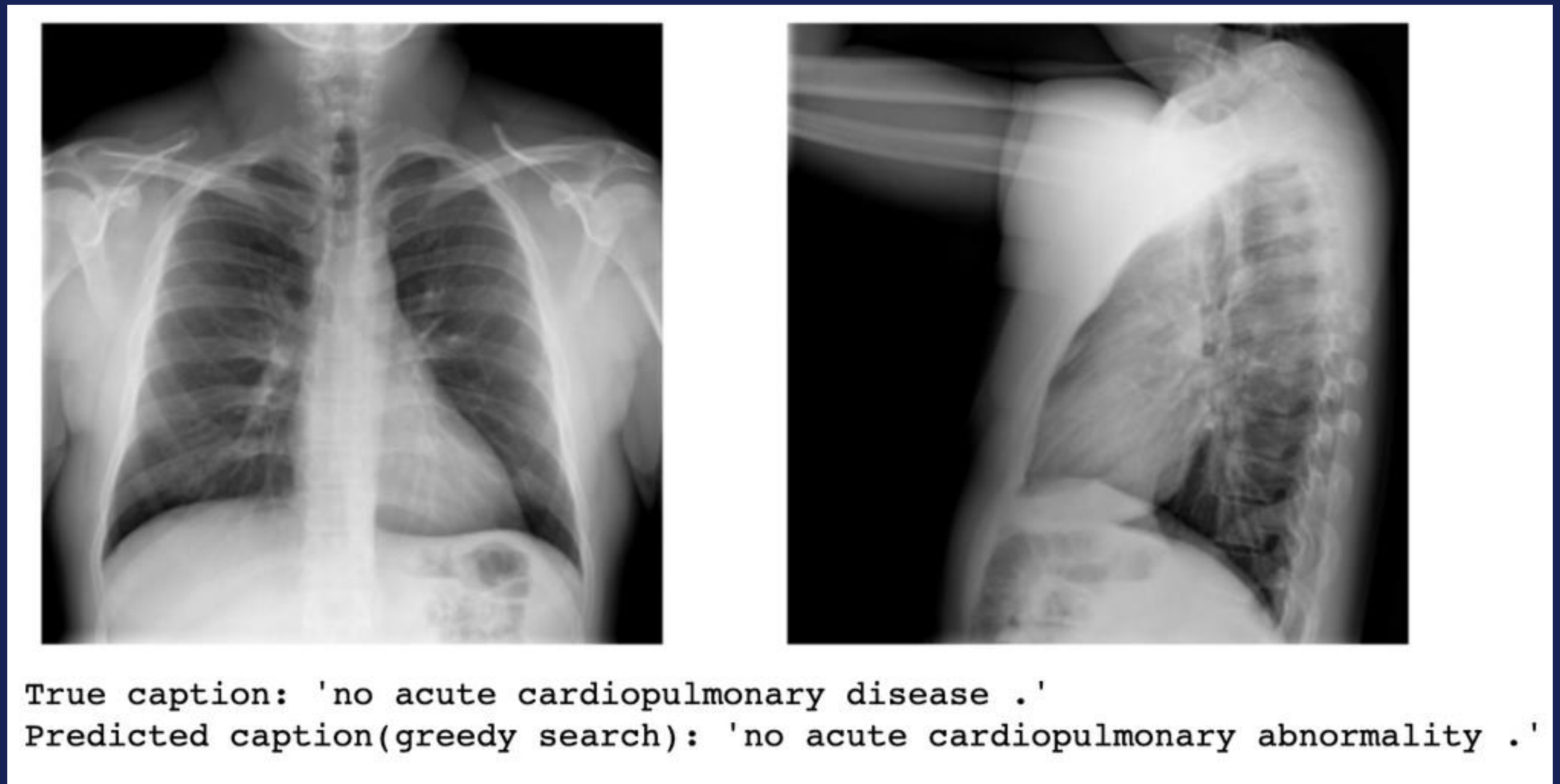
Image captioning can enhance CCTV systems by providing automated, realtime descriptions of captured scenes. This technology helps in identifying and reporting incidents .

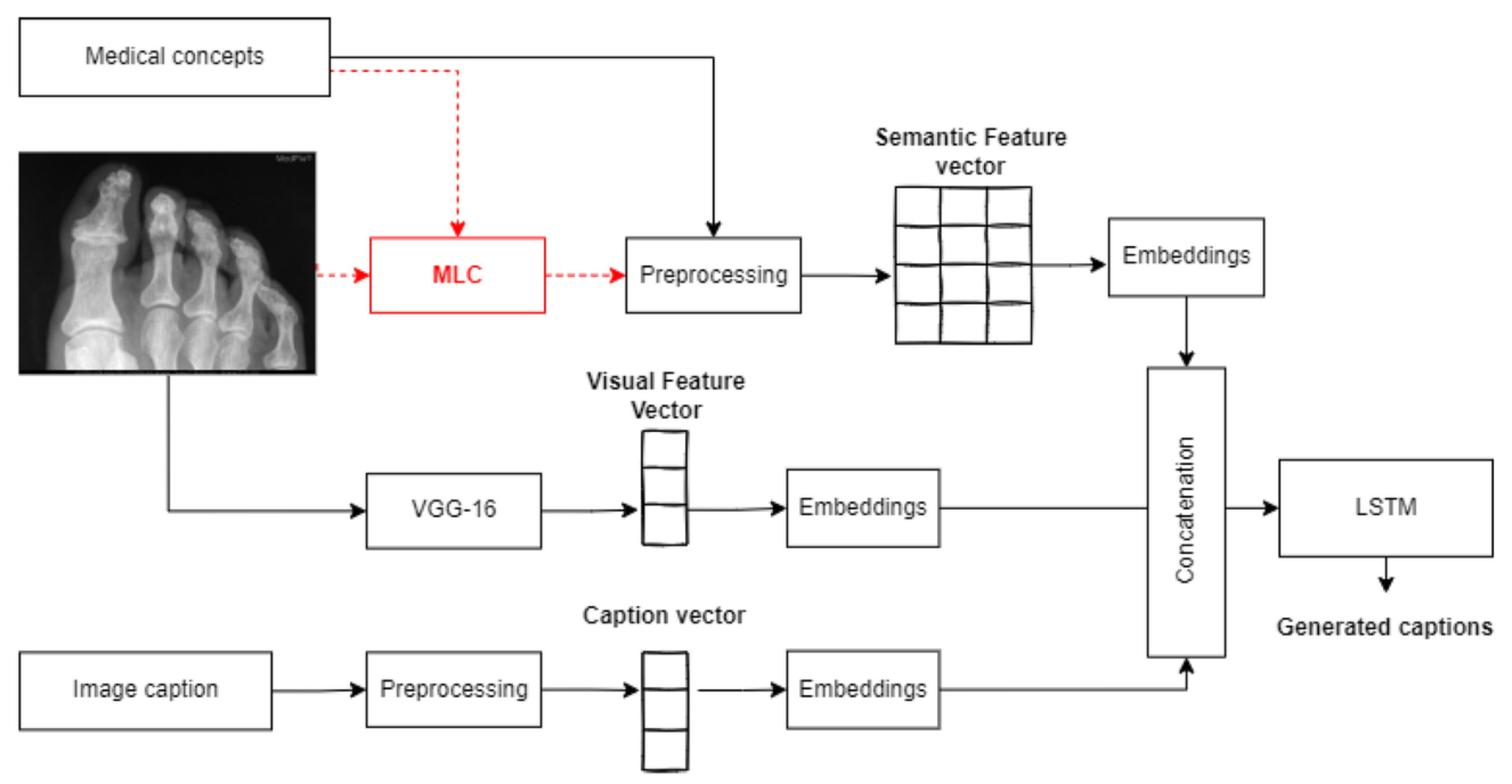
Destination Image Captioning

Travel Guide AI leverages image captioning to provide detailed descriptions and highlights of travel destinations.

DATASET

The ROCO dataset is a Medical images dataset that provides Radiology & Non Radiology image-caption pairs essential for training and evaluating the image captioning model. This dataset encompasses a variety of images, each associated with descriptive captions.





MODEL ARCHITECTURE

Combines a Convolutional Neural Network (CNN) and a LSTM for image captioning.Utilizes InceptionV3 for image feature extraction and LSTM for sequence processing.

INPUT LAYERS

Image Input: Shape (224, 224, 3) for 224x224 pixel RGB images.

Caption Input: Shape determined by the maximum caption length.

FEATURE EXTRACTOR (CNN)

The feature extractor (CNN) uses a pretrained InceptionV3 model with global average pooling to reduce spatial dimensions, keeping layers nontrainable to prevent overfitting. The output is then passed through a dense layer with 256 units and ReLU activation to produce the feature vector (fe2).

SEQUENCE PROCESSOR (LSTM):

The embedding layer maps words to 256-dimensional vectors with mask_zero=True. A dropout layer with a 0.5 rate prevents overfitting, followed by an LSTM layer with 256 units for sequence processing.

DECODER (COMBINING BOTH INPUTS)

The Add function combines CNN and LSTM outputs through elementwise addition. A dense layer with 256 units and ReLU activation (decoder2) is applied, followed by a final dense layer with softmax activation and vocab_size units to generate a probability distribution over the vocabulary.

MODEL COMPILATION

The model uses `categorical_crossentropy` as the loss function for multiclass classification, the `adam` optimizer for efficient training with an adaptive learning rate, and tracks accuracy as the performance metric during training and evaluation.

Model Summary

Layer (type)	Output Shape	Param #	Connected to
input_layer_2 (InputLayer)	(None, 78)	0	-
input_layer (InputLayer)	(None, 224, 224, 3)	0	-
embedding (Embedding)	(None, 78, 256)	740,864	input_layer_2[0]…
inception_v3 (Functional)	(None, 2048)	21,802,784	input_layer[0][0]
dropout (Dropout)	(None, 78, 256)	0	embedding[0][0]
not_equal (NotEqual)	(None, 78)	0	input_layer_2[0]…
dense (Dense)	(None, 256)	524,544	inception_v3[0][…
lstm (LSTM)	(None, 256)	525,312	dropout[0][0], not_equal[0][0]
add (Add)	(None, 256)	0	dense[0][0], lstm[0][0]
dense_1 (Dense)	(None, 256)	65,792	add[0][0]
dense_2 (Dense)	(None, 2894)	743,758	dense_1[0][0]

Total params: 24,403,054 (93.09 MB)

Trainable params: 2,600,270 (9.92 MB)

Non-trainable params: 21,802,784 (83.17 MB)

EPOCHS

```
[26]: history = model.fit([X1train, X2train], ytrain, epochs=epochs, batch_size=batch_size, validation_split=0.2)

Epoch 1/10
136/136 ━━━━━━━━━━━━━━━━━━━━ 27s 195ms/step - accuracy: 0.4696 - loss: 1.9253 - val_accuracy: 0.0633 - val_loss: 11.2858
Epoch 2/10
136/136 ━━━━━━━━━━━━━━━━━━━━ 24s 177ms/step - accuracy: 0.5385 - loss: 1.6388 - val_accuracy: 0.0536 - val_loss: 11.8283
Epoch 3/10
136/136 ━━━━━━━━━━━━━━━━━━━━ 23s 172ms/step - accuracy: 0.6177 - loss: 1.3423 - val_accuracy: 0.0573 - val_loss: 12.8448
Epoch 4/10
136/136 ━━━━━━━━━━━━━━━━━━━━ 23s 172ms/step - accuracy: 0.6682 - loss: 1.1504 - val_accuracy: 0.0578 - val_loss: 13.4812
Epoch 5/10
136/136 ━━━━━━━━━━━━━━━━━━━━ 24s 174ms/step - accuracy: 0.7480 - loss: 0.9083 - val_accuracy: 0.0536 - val_loss: 14.1454
Epoch 6/10
136/136 ━━━━━━━━━━━━━━━━━━━━ 24s 174ms/step - accuracy: 0.7866 - loss: 0.7547 - val_accuracy: 0.0546 - val_loss: 14.8854
Epoch 7/10
136/136 ━━━━━━━━━━━━━━━━━━━━ 24s 173ms/step - accuracy: 0.8428 - loss: 0.5880 - val_accuracy: 0.0490 - val_loss: 15.6709
Epoch 8/10
136/136 ━━━━━━━━━━━━━━━━━━━━ 24s 173ms/step - accuracy: 0.8737 - loss: 0.4848 - val_accuracy: 0.0513 - val_loss: 15.9703
Epoch 9/10
136/136 ━━━━━━━━━━━━━━━━━━━━ 24s 174ms/step - accuracy: 0.9063 - loss: 0.3817 - val_accuracy: 0.0541 - val_loss: 16.9059
Epoch 10/10
136/136 ━━━━━━━━━━━━━━━━━━━━ 24s 174ms/step - accuracy: 0.9309 - loss: 0.2932 - val_accuracy: 0.0601 - val_loss: 17.2857
```

MODEL EVALUATION

```
# Evaluate on training set
train_loss, train_acc = model.evaluate([X1train, X2train], ytrain, verbose=0)
print(f"Training Loss: {train_loss}, Training Accuracy: {train_acc}")
```

```
W0000 00:00:1721329459.541508     121 graph_launch.cc:671] Fallback to op-by-op mode because memset node breaks graph update
W0000 00:00:1721329495.893956     121 graph_launch.cc:671] Fallback to op-by-op mode because memset node breaks graph update
Training Loss: 3.5998265743255615, Training Accuracy: 0.7853906750679016
```

```
# Evaluate on test set
test_loss, test_acc = model.evaluate([X1test, X2test], ytest, verbose=0)
print(f"Test Loss: {test_loss}, Test Accuracy: {test_acc}")
```

```
Test Loss: 16.214902877807617, Test Accuracy: 0.059239938855171204
W0000 00:00:1721329526.476817     120 graph_launch.cc:671] Fallback to op-by-op mode because memset node breaks graph update
```

Fig : Model Evaluation on Train & Test sets

MODEL EVALUATION

```
Validation Accuracy: 0.05228758169934641
Validation Precision: 0.03325014842248419
Validation Recall: 0.05228758169934641
Validation F1 Score: 0.03743934672105607
```

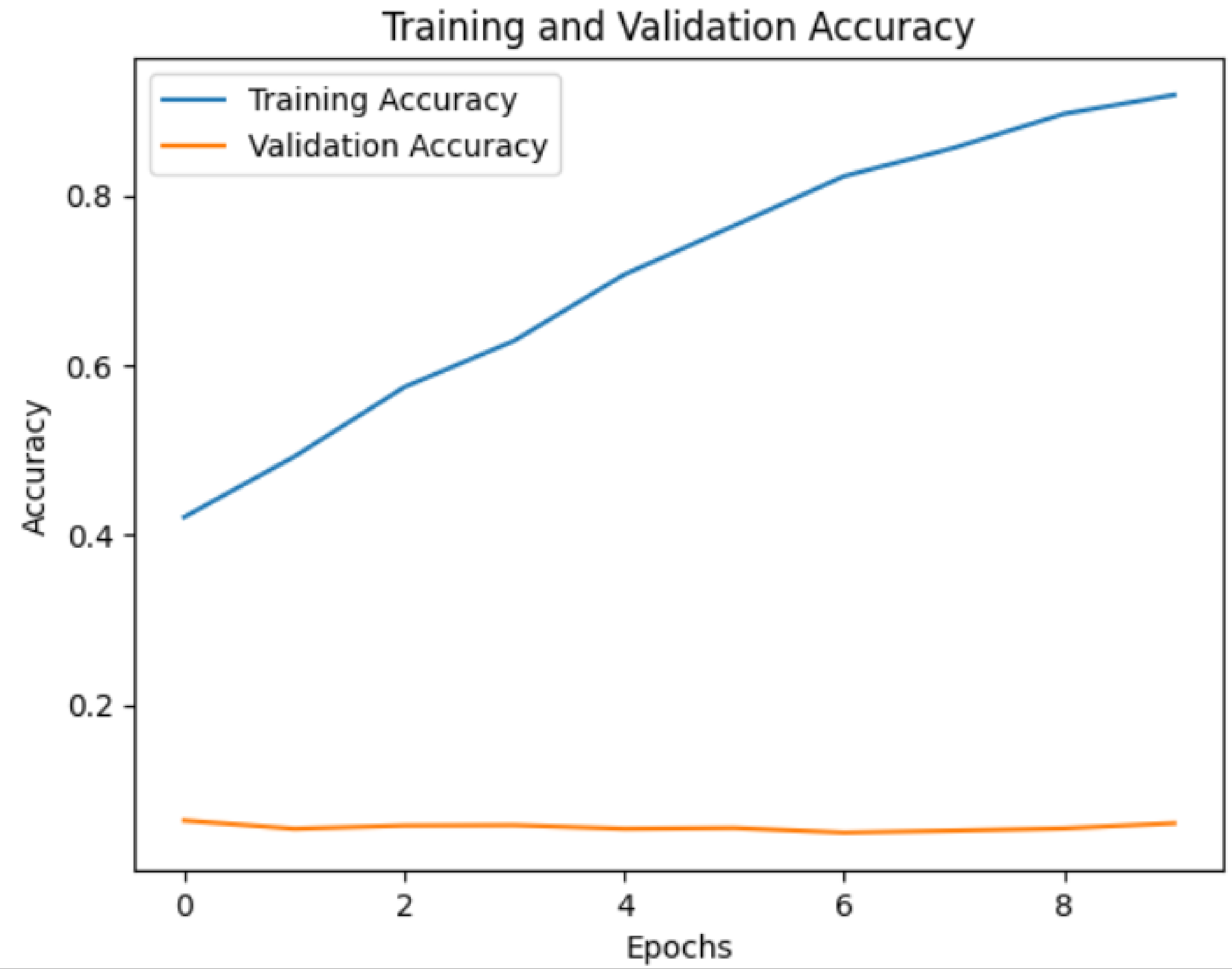


Fig : Model Evaluation on Validation Set & Plot

MODEL EVALUATION

```
Processed 1/20 images
Processed 2/20 images
Processed 3/20 images
Processed 4/20 images
Processed 5/20 images
Processed 6/20 images
Processed 7/20 images
Processed 8/20 images
Processed 9/20 images
Processed 10/20 images
Processed 11/20 images
Processed 12/20 images
Processed 13/20 images
Processed 14/20 images
Processed 15/20 images
Processed 16/20 images
Processed 17/20 images
Processed 18/20 images
Processed 19/20 images
Processed 20/20 images
Average BLEU score on validation set: 0.2079
```

Fig 4.8 : BLEU Score

CONCLUSION

The image captioning project demonstrated promising initial results, achieving a high training accuracy of 78.54%, which indicates the model's capability to learn and understand the training data effectively. Despite the challenges of overfitting and the need for better generalization to unseen data, the model's ability to generate meaningful captions is evident. The average BLEU score of 0.2079 on the validation set highlights its potential in generating reasonably aligned captions. With further refinements such as implementing regularization techniques, tuning hyperparameters, and enhancing the preprocessing pipeline, the model's performance can be significantly improved. These initial successes provide a strong foundation for continued development and optimization, ultimately leading to a robust image captioning system.

THANK YOU