Model Development Phase Template

Date	15 March 2024
Team ID	SWTID1720110092
Project Title	Beneath the Waves: Unravelling Coral Mysteries Through Deep Learning
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.

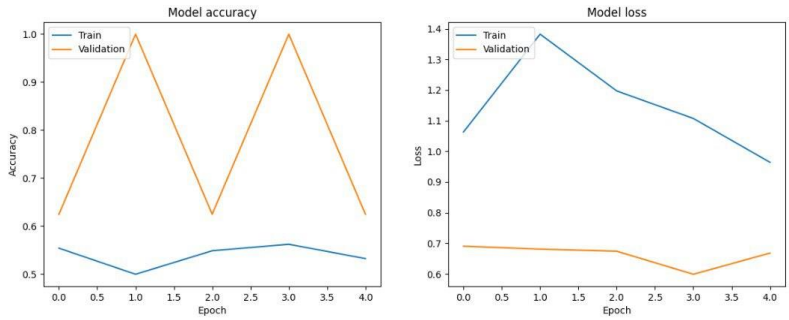
Initial Model Training Code (5 marks):

Paste the screenshot of the model training code

Model Validation and Evaluation Report (5 marks):

Model	Summary	Training and Validation Performance Metrics
Model 1	**Neural network summarized code**	import tensorflow as tf from tensorflow.keras.applications import EfficientNetB0 from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout, BatchNormalization from tensorflow.keras.models import Model from tensorflow.keras.optimizers import Adam

```
# Define the model
IMG_SIZE = (224, 224)
base_model = EfficientNetB0(weights='imagenet', include_top=False, input_shape=(*IMG_SIZE, 3))


base_model.trainable = True


fine_tune_at = 100
for layer in base_model.layers[:fine_tune_at]:
    layer.trainable = False


inputs = tf.keras.Input(shape=(*IMG_SIZE, 3))
x = base_model(inputs, training=True)
x = GlobalAveragePooling2D()(x)
x = BatchNormalization()(x)
x = Dropout(0.3)(x)
x = Dense(128, activation='relu')(x)
x = BatchNormalization()(x)
x = Dropout(0.3)(x)
outputs = Dense(2, activation='softmax')(x)


model = Model(inputs, outputs)
model.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_crossentropy', metrics=['accuracy'])


# Print the model summary
model.summary()
```

	Screenshot of the neural network summary	

	Training and Validation Code	import pandas as pd
		from tensorflow.keras.preprocessing.image import ImageDataGenerator
		
		# Load the data
		train_df = pd.read_csv('D:/train_updated.csv').sample(200)
		test_df = pd.read_csv('D:/test_updated.csv').sample(50)
		
		# Data augmentation and preprocessing
		train_datagen = ImageDataGenerator(
		rescale=1./255,
		rotation_range=20,
		width_shift_range=0.1,
		height_shift_range=0.1,
		shear_range=0.1,
		zoom_range=0.1,
		horizontal_flip=True,
		fill_mode='nearest'
		)
		
		test_datagen = ImageDataGenerator(rescale=1./255)

```
train_generator = train_datagen.flow_from_dataframe(
    dataframe=train_df,
    x_col='local_filename',
    y_col='label',
    target_size=IMG_SIZE,
    batch_size=16,
    class_mode='categorical'
)

test_generator = test_datagen.flow_from_dataframe(
    dataframe=test_df,
    x_col='local_filename',
    y_col='label',
    target_size=IMG_SIZE,
    batch_size=16,
    class_mode='categorical',
    shuffle=False
)

# Train the model
history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // 16,
    epochs=5,
    validation_data=test_generator,
```

```
    validation_steps=test_generator.samples // 16
)


# Save the model
model.save('coral_classification_model.h5')


# Output training and validation performance metrics
import matplotlib.pyplot as plt


# Plot training & validation accuracy values
plt.figure(figsize=(14, 5))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')


# Plot training & validation loss values
plt.subplot(1, 2, 2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
```

		plt.legend(['Train', 'Validation'], loc='upper left')
		
		# Save the plot as a file
		plt.savefig('training_validation_metrics.png')
		
		# Show the plots
		plt.show()