Untitled3

July 27, 2024

[7]:
```
import os
import numpy as np
import cv2
from glob import glob
import tensorflow as tf
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split

from tensorflow.keras.layers import Conv2D, Activation, BatchNormalization,␣
↪UpSampling2D, Input, Concatenate, Cropping2D
from tensorflow.keras.models import Model
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
from tensorflow.keras import backend as K

# Set NumPy random seed for reproducibility
np.random.seed(42)

# Set TensorFlow random seed for reproducibility
tf.random.set_seed(42)

# Define image size for resizing input images and masks
WIDTH = 672
HEIGHT = 504

# Define number of epochs for training
EPOCHS = 10

# Define batch size
BATCH = 16

# Define learning rate for the optimizer
LR = 1e-4

# Define output path for saving training results and model
PATH = "/root/61541v001/V-03"
```

```
# Define label colors
colors = [
    (0, 0, 0), (128, 0, 0), (0, 128, 0), (128, 128, 0), (0, 0, 128),
    (128, 0, 128), (0, 128, 128), (128, 128, 128), (64, 0, 0), (192, 0, 0),
    (64, 128, 0), (192, 128, 0), (64, 0, 128), (192, 0, 128), (64, 128, 128),
    (192, 128, 128), (0, 64, 0), (128, 64, 0), (0, 192, 0), (128, 192, 0),
    (0, 64, 128), (128, 64, 12)
]
NUM_CLASSES = len(colors)
```

[8]:
```
# Load dataset
def load_data(path, split=0.1):
    images = sorted(glob(os.path.join(PATH, "image/*")))
    masks = sorted(glob(os.path.join(PATH, "label/*")))
    total_size = len(images)
    valid_size = int(split * total_size)
    test_size = int(split * total_size)

    train_x, valid_x = train_test_split(images, test_size=valid_size,␣
 ↪random_state=42)
    train_y, valid_y = train_test_split(masks, test_size=valid_size,␣
 ↪random_state=42)
    train_x, test_x = train_test_split(train_x, test_size=test_size,␣
 ↪random_state=42)
    train_y, test_y = train_test_split(train_y, test_size=test_size,␣
 ↪random_state=42)

    return (train_x, train_y), (valid_x, valid_y), (test_x, test_y)

# Read and preprocess image
def read_image(path):
    if isinstance(path, bytes):
        path = path.decode()
    x = cv2.imread(path, cv2.IMREAD_COLOR)
    x = cv2.cvtColor(x, cv2.COLOR_BGR2RGB)
    x = cv2.resize(x, (WIDTH, HEIGHT))
    x = x / 255.0
    return x.astype(np.float32)

# Read and preprocess mask
def read_mask(path):
    if isinstance(path, bytes):
        path = path.decode()
    x = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
    x = cv2.resize(x, (WIDTH, HEIGHT))
    x = x / 255.0
    x = np.round(x * (NUM_CLASSES - 1)).astype(np.uint8)
```

```
    x = tf.keras.utils.to_categorical(x, num_classes=NUM_CLASSES)
    return x.astype(np.float32)

# Parse function for TensorFlow dataset
def tf_parse(x, y):
    def _parse(x, y):
        x = read_image(x)
        y = read_mask(y)
        return x, y

    x, y = tf.numpy_function(_parse, [x, y], [tf.float32, tf.float32])
    x.set_shape([HEIGHT, WIDTH, 3])
    y.set_shape([HEIGHT, WIDTH, NUM_CLASSES])
    return x, y

# Create TensorFlow dataset
def tf_dataset(x, y, batch=8):
    dataset = tf.data.Dataset.from_tensor_slices((x, y))
    dataset = dataset.map(tf_parse)
    dataset = dataset.batch(batch)
    dataset = dataset.repeat()
    return dataset

# Parse mask for visualization
def mask_parse(mask):
    mask = np.argmax(mask, axis=-1)  # Convert one-hot encoded mask to␣
 ↪single-channel class mask
    rgb_mask = np.zeros((mask.shape[0], mask.shape[1], 3), dtype=np.uint8)  #␣
 ↪Create an empty color mask
    for i, color in enumerate(colors):
        rgb_mask[mask == i] = color  # Map class values to predefined colors
    return rgb_mask

# Build the model
def build_model():
    inputs = Input(shape=(HEIGHT, WIDTH, 3), name="input_image")
    encoder = MobileNetV2(input_tensor=inputs, weights="imagenet",␣
 ↪include_top=False, alpha=0.35)
    skip_connection_names = ["input_image", "block_1_expand_relu",␣
 ↪"block_3_expand_relu", "block_6_expand_relu"]
    encoder_output = encoder.get_layer("block_13_expand_relu").output
    f = [16, 32, 48, 64]
    x = encoder_output

    for i in range(1, len(skip_connection_names) + 1, 1):
        x_skip = encoder.get_layer(skip_connection_names[-i]).output
        x = UpSampling2D((2, 2))(x)
```

```
        height_diff = K.int_shape(x)[1] - K.int_shape(x_skip)[1]
        width_diff = K.int_shape(x)[2] - K.int_shape(x_skip)[2]
        if height_diff != 0 or width_diff != 0:
            x = Cropping2D(((height_diff // 2, height_diff - height_diff // 2),
                            (width_diff // 2, width_diff - width_diff // 2)))(x)
        x = Concatenate()([x, x_skip])
        x = Conv2D(f[-i], (3, 3), padding="same")(x)
        x = BatchNormalization()(x)
        x = Activation("relu")(x)
        x = Conv2D(f[-i], (3, 3), padding="same")(x)
        x = BatchNormalization()(x)
        x = Activation("relu")(x)

    x = Conv2D(NUM_CLASSES, (1, 1), padding="same")(x)
    x = Activation("softmax")(x)
    model = Model(inputs, x)
    return model
```

[9]:
```
# Load dataset
(train_x, train_y), (valid_x, valid_y), (test_x, test_y) = load_data(PATH)
print("Training data: ", len(train_x))
print("Validation data: ", len(valid_x))
print("Testing data: ", len(test_x))
```

```
Training data:  1477
Validation data:  184
Testing data:  184
```

[10]:
```
# Define custom metrics
def dice_coefficient(y_true, y_pred):
    y_true_f = tf.keras.backend.flatten(y_true)
    y_pred_f = tf.keras.backend.flatten(y_pred)
    intersection = tf.keras.backend.sum(y_true_f * y_pred_f)
    return (2. * intersection) / (tf.keras.backend.sum(y_true_f) + tf.keras.
 ↪backend.sum(y_pred_f))

def iou(y_true, y_pred):
    y_true_f = tf.keras.backend.flatten(y_true)
    y_pred_f = tf.keras.backend.flatten(y_pred)
    intersection = tf.keras.backend.sum(y_true_f * y_pred_f)
    union = tf.keras.backend.sum(y_true_f) + tf.keras.backend.sum(y_pred_f) -␣
 ↪intersection
    return intersection / union
```

[11]:
```
# Create and compile the model
model = build_model()
model.summary()
```

```
loss = tf.keras.losses.CategoricalCrossentropy()
opt = tf.keras.optimizers.Nadam(LR)
metrics = ['accuracy']
model.compile(loss=loss, optimizer=opt, metrics=['accuracy', dice_coefficient,␣
 ↪iou])
```

```
/tmp/ipykernel_5823/2080664991.py:68: UserWarning: `input_shape` is undefined or
non-square, or `rows` is not in [96, 128, 160, 192, 224]. Weights for input
shape (224, 224) will be loaded as the default.
  encoder = MobileNetV2(input_tensor=inputs, weights="imagenet",
include_top=False, alpha=0.35)
```

Model: "functional_1"

Layer (type)	Output Shape	Param #	Connected to
input_image (InputLayer)	(None, 504, 672, 3)	0	-
Conv1 (Conv2D)	(None, 252, 336, 16)	432	input_image[0][0]
bn_Conv1 (BatchNormalizatio…	(None, 252, 336, 16)	64	Conv1[0][0]
Conv1_relu (ReLU)	(None, 252, 336, 16)	0	bn_Conv1[0][0]
expanded_conv_dept… (DepthwiseConv2D)	(None, 252, 336, 16)	144	Conv1_relu[0][0]
expanded_conv_dept… (BatchNormalizatio…	(None, 252, 336, 16)	64	expanded_conv_de…
expanded_conv_dept… (ReLU)	(None, 252, 336, 16)	0	expanded_conv_de…
expanded_conv_proj… (Conv2D)	(None, 252, 336, 8)	128	expanded_conv_de…
expanded_conv_proj… (BatchNormalizatio…	(None, 252, 336, 8)	32	expanded_conv_pr…
block_1_expand (Conv2D)	(None, 252, 336, 48)	384	expanded_conv_pr…

```
block_1_expand_BN     (None, 252, 336,          192   block_1_expand[0…
(BatchNormalizatio…   48)

block_1_expand_relu   (None, 252, 336,            0   block_1_expand_B…
(ReLU)                48)

block_1_pad           (None, 253, 337,            0   block_1_expand_r…
(ZeroPadding2D)       48)

block_1_depthwise     (None, 126, 168,          432   block_1_pad[0][0]
(DepthwiseConv2D)     48)

block_1_depthwise_…   (None, 126, 168,          192   block_1_depthwis…
(BatchNormalizatio…   48)

block_1_depthwise_…   (None, 126, 168,            0   block_1_depthwis…
(ReLU)                48)

block_1_project       (None, 126, 168,          384   block_1_depthwis…
(Conv2D)              8)

block_1_project_BN    (None, 126, 168,           32   block_1_project[…
(BatchNormalizatio…   8)

block_2_expand        (None, 126, 168,          384   block_1_project_…
(Conv2D)              48)

block_2_expand_BN     (None, 126, 168,          192   block_2_expand[0…
(BatchNormalizatio…   48)

block_2_expand_relu   (None, 126, 168,            0   block_2_expand_B…
(ReLU)                48)

block_2_depthwise     (None, 126, 168,          432   block_2_expand_r…
(DepthwiseConv2D)     48)

block_2_depthwise_…   (None, 126, 168,          192   block_2_depthwis…
(BatchNormalizatio…   48)

block_2_depthwise_…   (None, 126, 168,            0   block_2_depthwis…
(ReLU)                48)

block_2_project       (None, 126, 168,          384   block_2_depthwis…
(Conv2D)              8)

block_2_project_BN    (None, 126, 168,           32   block_2_project[…
(BatchNormalizatio…   8)
```

block_2_add (Add)	(None, 126, 168, 8)	0	block_1_project_… block_2_project_…
block_3_expand (Conv2D)	(None, 126, 168, 48)	384	block_2_add[0][0]
block_3_expand_BN (BatchNormalizatio…	(None, 126, 168, 48)	192	block_3_expand[0…
block_3_expand_relu (ReLU)	(None, 126, 168, 48)	0	block_3_expand_B…
block_3_pad (ZeroPadding2D)	(None, 127, 169, 48)	0	block_3_expand_r…
block_3_depthwise (DepthwiseConv2D)	(None, 63, 84, 48)	432	block_3_pad[0][0]
block_3_depthwise_… (BatchNormalizatio…	(None, 63, 84, 48)	192	block_3_depthwis…
block_3_depthwise_… (ReLU)	(None, 63, 84, 48)	0	block_3_depthwis…
block_3_project (Conv2D)	(None, 63, 84, 16)	768	block_3_depthwis…
block_3_project_BN (BatchNormalizatio…	(None, 63, 84, 16)	64	block_3_project[…
block_4_expand (Conv2D)	(None, 63, 84, 96)	1,536	block_3_project_…
block_4_expand_BN (BatchNormalizatio…	(None, 63, 84, 96)	384	block_4_expand[0…
block_4_expand_relu (ReLU)	(None, 63, 84, 96)	0	block_4_expand_B…
block_4_depthwise (DepthwiseConv2D)	(None, 63, 84, 96)	864	block_4_expand_r…
block_4_depthwise_… (BatchNormalizatio…	(None, 63, 84, 96)	384	block_4_depthwis…
block_4_depthwise_… (ReLU)	(None, 63, 84, 96)	0	block_4_depthwis…

block_4_project (Conv2D)	(None, 63, 84, 16)	1,536	block_4_depthwis…
block_4_project_BN (BatchNormalizatio…	(None, 63, 84, 16)	64	block_4_project[…
block_4_add (Add)	(None, 63, 84, 16)	0	block_3_project_… block_4_project_…
block_5_expand (Conv2D)	(None, 63, 84, 96)	1,536	block_4_add[0][0]
block_5_expand_BN (BatchNormalizatio…	(None, 63, 84, 96)	384	block_5_expand[0…
block_5_expand_relu (ReLU)	(None, 63, 84, 96)	0	block_5_expand_B…
block_5_depthwise (DepthwiseConv2D)	(None, 63, 84, 96)	864	block_5_expand_r…
block_5_depthwise_… (BatchNormalizatio…	(None, 63, 84, 96)	384	block_5_depthwis…
block_5_depthwise_… (ReLU)	(None, 63, 84, 96)	0	block_5_depthwis…
block_5_project (Conv2D)	(None, 63, 84, 16)	1,536	block_5_depthwis…
block_5_project_BN (BatchNormalizatio…	(None, 63, 84, 16)	64	block_5_project[…
block_5_add (Add)	(None, 63, 84, 16)	0	block_4_add[0][0… block_5_project_…
block_6_expand (Conv2D)	(None, 63, 84, 96)	1,536	block_5_add[0][0]
block_6_expand_BN (BatchNormalizatio…	(None, 63, 84, 96)	384	block_6_expand[0…
block_6_expand_relu (ReLU)	(None, 63, 84, 96)	0	block_6_expand_B…
block_6_pad (ZeroPadding2D)	(None, 65, 85, 96)	0	block_6_expand_r…

block_6_depthwise (DepthwiseConv2D)	(None, 32, 42, 96)	864	block_6_pad[0][0]
block_6_depthwise_… (BatchNormalizatio…	(None, 32, 42, 96)	384	block_6_depthwis…
block_6_depthwise_… (ReLU)	(None, 32, 42, 96)	0	block_6_depthwis…
block_6_project (Conv2D)	(None, 32, 42, 24)	2,304	block_6_depthwis…
block_6_project_BN (BatchNormalizatio…	(None, 32, 42, 24)	96	block_6_project[…
block_7_expand (Conv2D)	(None, 32, 42, 144)	3,456	block_6_project_…
block_7_expand_BN (BatchNormalizatio…	(None, 32, 42, 144)	576	block_7_expand[0…
block_7_expand_relu (ReLU)	(None, 32, 42, 144)	0	block_7_expand_B…
block_7_depthwise (DepthwiseConv2D)	(None, 32, 42, 144)	1,296	block_7_expand_r…
block_7_depthwise_… (BatchNormalizatio…	(None, 32, 42, 144)	576	block_7_depthwis…
block_7_depthwise_… (ReLU)	(None, 32, 42, 144)	0	block_7_depthwis…
block_7_project (Conv2D)	(None, 32, 42, 24)	3,456	block_7_depthwis…
block_7_project_BN (BatchNormalizatio…	(None, 32, 42, 24)	96	block_7_project[…
block_7_add (Add)	(None, 32, 42, 24)	0	block_6_project_… block_7_project_…
block_8_expand (Conv2D)	(None, 32, 42, 144)	3,456	block_7_add[0][0]
block_8_expand_BN (BatchNormalizatio…	(None, 32, 42, 144)	576	block_8_expand[0…

block_8_expand_relu (ReLU)	(None, 32, 42, 144)	0	block_8_expand_B…
block_8_depthwise (DepthwiseConv2D)	(None, 32, 42, 144)	1,296	block_8_expand_r…
block_8_depthwise_… (BatchNormalizatio…	(None, 32, 42, 144)	576	block_8_depthwis…
block_8_depthwise_… (ReLU)	(None, 32, 42, 144)	0	block_8_depthwis…
block_8_project (Conv2D)	(None, 32, 42, 24)	3,456	block_8_depthwis…
block_8_project_BN (BatchNormalizatio…	(None, 32, 42, 24)	96	block_8_project[…
block_8_add (Add)	(None, 32, 42, 24)	0	block_7_add[0][0… block_8_project_…
block_9_expand (Conv2D)	(None, 32, 42, 144)	3,456	block_8_add[0][0]
block_9_expand_BN (BatchNormalizatio…	(None, 32, 42, 144)	576	block_9_expand[0…
block_9_expand_relu (ReLU)	(None, 32, 42, 144)	0	block_9_expand_B…
block_9_depthwise (DepthwiseConv2D)	(None, 32, 42, 144)	1,296	block_9_expand_r…
block_9_depthwise_… (BatchNormalizatio…	(None, 32, 42, 144)	576	block_9_depthwis…
block_9_depthwise_… (ReLU)	(None, 32, 42, 144)	0	block_9_depthwis…
block_9_project (Conv2D)	(None, 32, 42, 24)	3,456	block_9_depthwis…
block_9_project_BN (BatchNormalizatio…	(None, 32, 42, 24)	96	block_9_project[…
block_9_add (Add)	(None, 32, 42, 24)	0	block_8_add[0][0… block_9_project_…

block_10_expand (Conv2D)	(None, 32, 42, 144)	3,456	block_9_add[0][0]
block_10_expand_BN (BatchNormalizatio…	(None, 32, 42, 144)	576	block_10_expand[…
block_10_expand_re… (ReLU)	(None, 32, 42, 144)	0	block_10_expand_…
block_10_depthwise (DepthwiseConv2D)	(None, 32, 42, 144)	1,296	block_10_expand_…
block_10_depthwise… (BatchNormalizatio…	(None, 32, 42, 144)	576	block_10_depthwi…
block_10_depthwise… (ReLU)	(None, 32, 42, 144)	0	block_10_depthwi…
block_10_project (Conv2D)	(None, 32, 42, 32)	4,608	block_10_depthwi…
block_10_project_BN (BatchNormalizatio…	(None, 32, 42, 32)	128	block_10_project…
block_11_expand (Conv2D)	(None, 32, 42, 192)	6,144	block_10_project…
block_11_expand_BN (BatchNormalizatio…	(None, 32, 42, 192)	768	block_11_expand[…
block_11_expand_re… (ReLU)	(None, 32, 42, 192)	0	block_11_expand_…
block_11_depthwise (DepthwiseConv2D)	(None, 32, 42, 192)	1,728	block_11_expand_…
block_11_depthwise… (BatchNormalizatio…	(None, 32, 42, 192)	768	block_11_depthwi…
block_11_depthwise… (ReLU)	(None, 32, 42, 192)	0	block_11_depthwi…
block_11_project (Conv2D)	(None, 32, 42, 32)	6,144	block_11_depthwi…
block_11_project_BN (BatchNormalizatio…	(None, 32, 42, 32)	128	block_11_project…

block_11_add (Add)	(None, 32, 42, 32)	0	block_10_project… block_11_project…
block_12_expand (Conv2D)	(None, 32, 42, 192)	6,144	block_11_add[0][…
block_12_expand_BN (BatchNormalizatio…	(None, 32, 42, 192)	768	block_12_expand[…
block_12_expand_re… (ReLU)	(None, 32, 42, 192)	0	block_12_expand_…
block_12_depthwise (DepthwiseConv2D)	(None, 32, 42, 192)	1,728	block_12_expand_…
block_12_depthwise… (BatchNormalizatio…	(None, 32, 42, 192)	768	block_12_depthwi…
block_12_depthwise… (ReLU)	(None, 32, 42, 192)	0	block_12_depthwi…
block_12_project (Conv2D)	(None, 32, 42, 32)	6,144	block_12_depthwi…
block_12_project_BN (BatchNormalizatio…	(None, 32, 42, 32)	128	block_12_project…
block_12_add (Add)	(None, 32, 42, 32)	0	block_11_add[0][… block_12_project…
block_13_expand (Conv2D)	(None, 32, 42, 192)	6,144	block_12_add[0][…
block_13_expand_BN (BatchNormalizatio…	(None, 32, 42, 192)	768	block_13_expand[…
block_13_expand_re… (ReLU)	(None, 32, 42, 192)	0	block_13_expand_…
up_sampling2d_4 (UpSampling2D)	(None, 64, 84, 192)	0	block_13_expand_…
cropping2d_1 (Cropping2D)	(None, 63, 84, 192)	0	up_sampling2d_4[…
concatenate_4 (Concatenate)	(None, 63, 84, 288)	0	cropping2d_1[0][… block_6_expand_r…

conv2d_9 (Conv2D)	(None, 63, 84, 64)	165,952	concatenate_4[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 63, 84, 64)	256	conv2d_9[0][0]
activation_9 (Activation)	(None, 63, 84, 64)	0	batch_normalizat…
conv2d_10 (Conv2D)	(None, 63, 84, 64)	36,928	activation_9[0][…
batch_normalizatio… (BatchNormalizatio…	(None, 63, 84, 64)	256	conv2d_10[0][0]
activation_10 (Activation)	(None, 63, 84, 64)	0	batch_normalizat…
up_sampling2d_5 (UpSampling2D)	(None, 126, 168, 64)	0	activation_10[0]…
concatenate_5 (Concatenate)	(None, 126, 168, 112)	0	up_sampling2d_5[… block_3_expand_r…
conv2d_11 (Conv2D)	(None, 126, 168, 48)	48,432	concatenate_5[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 126, 168, 48)	192	conv2d_11[0][0]
activation_11 (Activation)	(None, 126, 168, 48)	0	batch_normalizat…
conv2d_12 (Conv2D)	(None, 126, 168, 48)	20,784	activation_11[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 126, 168, 48)	192	conv2d_12[0][0]
activation_12 (Activation)	(None, 126, 168, 48)	0	batch_normalizat…
up_sampling2d_6 (UpSampling2D)	(None, 252, 336, 48)	0	activation_12[0]…
concatenate_6 (Concatenate)	(None, 252, 336, 96)	0	up_sampling2d_6[… block_1_expand_r…

conv2d_13 (Conv2D)	(None, 252, 336, 32)	27,680	concatenate_6[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 252, 336, 32)	128	conv2d_13[0][0]
activation_13 (Activation)	(None, 252, 336, 32)	0	batch_normalizat…
conv2d_14 (Conv2D)	(None, 252, 336, 32)	9,248	activation_13[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 252, 336, 32)	128	conv2d_14[0][0]
activation_14 (Activation)	(None, 252, 336, 32)	0	batch_normalizat…
up_sampling2d_7 (UpSampling2D)	(None, 504, 672, 32)	0	activation_14[0]…
concatenate_7 (Concatenate)	(None, 504, 672, 35)	0	up_sampling2d_7[… input_image[0][0]
conv2d_15 (Conv2D)	(None, 504, 672, 16)	5,056	concatenate_7[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 504, 672, 16)	64	conv2d_15[0][0]
activation_15 (Activation)	(None, 504, 672, 16)	0	batch_normalizat…
conv2d_16 (Conv2D)	(None, 504, 672, 16)	2,320	activation_15[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 504, 672, 16)	64	conv2d_16[0][0]
activation_16 (Activation)	(None, 504, 672, 16)	0	batch_normalizat…
conv2d_17 (Conv2D)	(None, 504, 672, 22)	374	activation_16[0]…
activation_17 (Activation)	(None, 504, 672, 22)	0	conv2d_17[0][0]

Total params: 416,566 (1.59 MB)

Trainable params: 409,382 (1.56 MB)

Non-trainable params: 7,184 (28.06 KB)

[12]:
```
# Define callbacks
callbacks = [
    ReduceLROnPlateau(monitor='val_loss', factor=0.1, patience=4),
    EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=False)
]

train_steps = len(train_x) // BATCH
valid_steps = len(valid_x) // BATCH
if len(train_x) % BATCH != 0:
    train_steps += 1
if len(valid_x) % BATCH != 0:
    valid_steps += 1

train_dataset = tf_dataset(train_x, train_y, batch=BATCH)
valid_dataset = tf_dataset(valid_x, valid_y, batch=BATCH)

# Train the model and record history
history = model.fit(
    train_dataset,
    validation_data=valid_dataset,
    epochs=EPOCHS,
    steps_per_epoch=train_steps,
    validation_steps=valid_steps,
    callbacks=callbacks
)
```

```
Epoch 1/10
93/93             158s 2s/step -
accuracy: 0.1966 - dice_coefficient: 0.0808 - iou: 0.0424 - loss: 2.8047 -
val_accuracy: 0.6564 - val_dice_coefficient: 0.1105 - val_iou: 0.0585 -
val_loss: 2.4065 - learning_rate: 1.0000e-04
Epoch 2/10
93/93             123s 1s/step -
accuracy: 0.6876 - dice_coefficient: 0.1673 - iou: 0.0913 - loss: 2.0708 -
val_accuracy: 0.7447 - val_dice_coefficient: 0.2234 - val_iou: 0.1258 -
val_loss: 1.8142 - learning_rate: 1.0000e-04
Epoch 3/10
93/93             116s 1s/step -
```

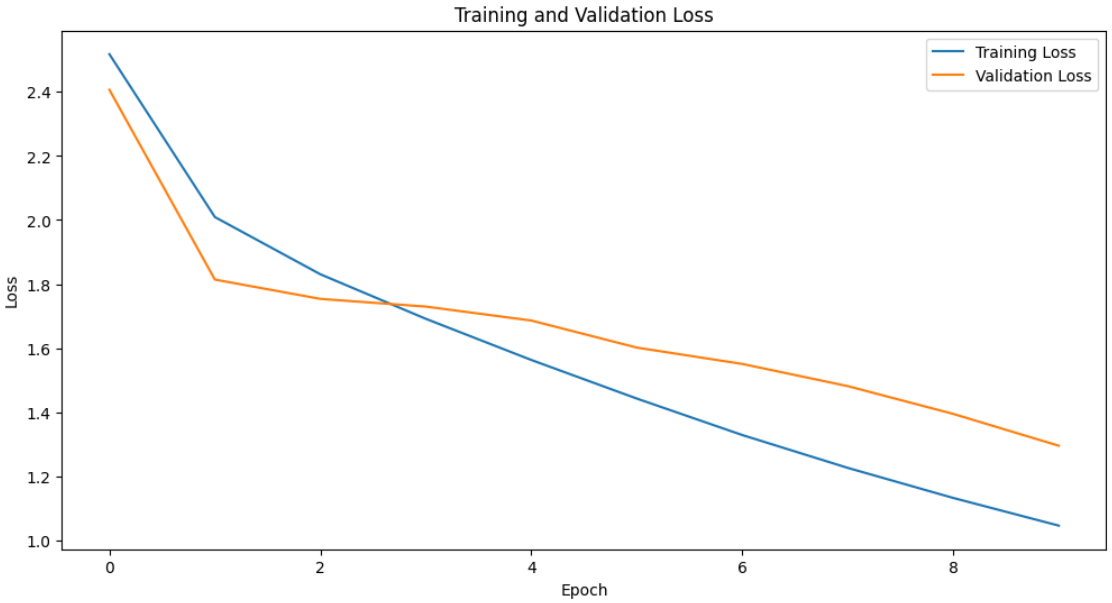
```
accuracy: 0.7347 - dice_coefficient: 0.2003 - iou: 0.1113 - loss: 1.8708 -
val_accuracy: 0.7605 - val_dice_coefficient: 0.2319 - val_iou: 0.1312 -
val_loss: 1.7540 - learning_rate: 1.0000e-04
Epoch 4/10
93/93            112s 1s/step -
accuracy: 0.7511 - dice_coefficient: 0.2293 - iou: 0.1295 - loss: 1.7280 -
val_accuracy: 0.7689 - val_dice_coefficient: 0.2366 - val_iou: 0.1342 -
val_loss: 1.7300 - learning_rate: 1.0000e-04
Epoch 5/10
93/93            115s 1s/step -
accuracy: 0.7700 - dice_coefficient: 0.2586 - iou: 0.1485 - loss: 1.5982 -
val_accuracy: 0.7793 - val_dice_coefficient: 0.2496 - val_iou: 0.1426 -
val_loss: 1.6864 - learning_rate: 1.0000e-04
Epoch 6/10
93/93            114s 1s/step -
accuracy: 0.8451 - dice_coefficient: 0.2892 - iou: 0.1691 - loss: 1.4758 -
val_accuracy: 0.7628 - val_dice_coefficient: 0.2850 - val_iou: 0.1662 -
val_loss: 1.6020 - learning_rate: 1.0000e-04
Epoch 7/10
93/93            115s 1s/step -
accuracy: 0.8481 - dice_coefficient: 0.3228 - iou: 0.1925 - loss: 1.3600 -
val_accuracy: 0.7501 - val_dice_coefficient: 0.3067 - val_iou: 0.1811 -
val_loss: 1.5510 - learning_rate: 1.0000e-04
Epoch 8/10
93/93            113s 1s/step -
accuracy: 0.8499 - dice_coefficient: 0.3562 - iou: 0.2167 - loss: 1.2549 -
val_accuracy: 0.7478 - val_dice_coefficient: 0.3284 - val_iou: 0.1965 -
val_loss: 1.4818 - learning_rate: 1.0000e-04
Epoch 9/10
93/93            116s 1s/step -
accuracy: 0.8517 - dice_coefficient: 0.3894 - iou: 0.2419 - loss: 1.1591 -
val_accuracy: 0.7505 - val_dice_coefficient: 0.3562 - val_iou: 0.2167 -
val_loss: 1.3951 - learning_rate: 1.0000e-04
Epoch 10/10
93/93            115s 1s/step -
accuracy: 0.8535 - dice_coefficient: 0.4228 - iou: 0.2681 - loss: 1.0710 -
val_accuracy: 0.7596 - val_dice_coefficient: 0.3874 - val_iou: 0.2403 -
val_loss: 1.2958 - learning_rate: 1.0000e-04
```

[13]:
```python
# Evaluate the model
test_dataset = tf_dataset(test_x, test_y, batch=BATCH)
test_steps = (len(test_x) // BATCH)
if len(test_x) % BATCH != 0:
    test_steps += 1
results = model.evaluate(test_dataset, steps=test_steps)

# Get metric names
```

```
metrics_names = model.metrics_names

# Print evaluation results
for name, value in zip(metrics_names, results):
    print(f"{name}: {value:.4f}")
```

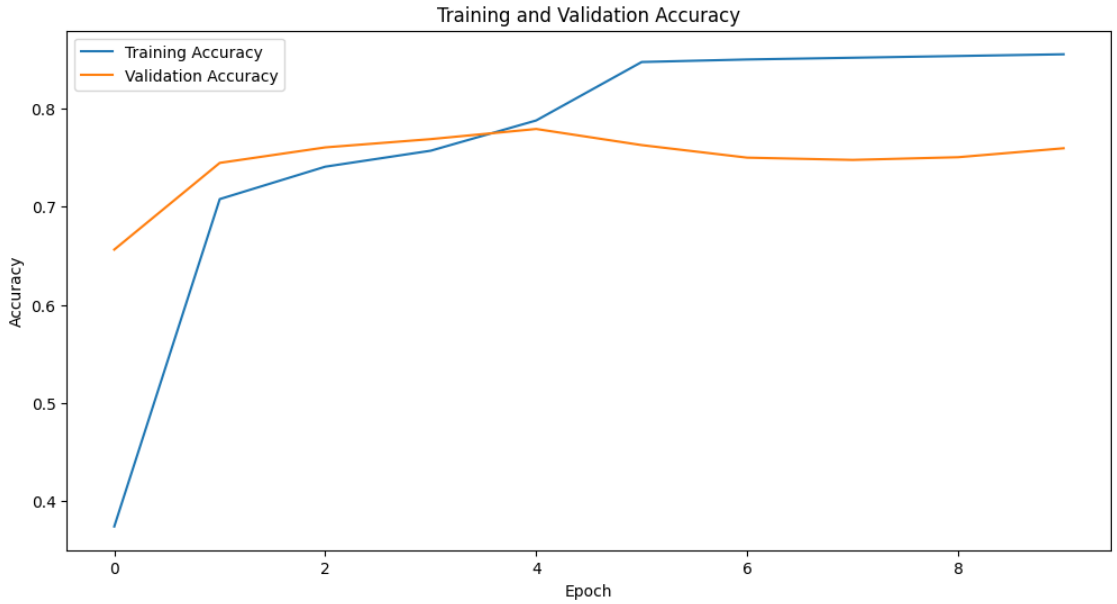
```
12/12              14s 1s/step -
accuracy: 0.7515 - dice_coefficient: 0.3826 - iou: 0.2366 - loss: 1.3126
loss: 1.3212
compile_metrics: 0.7492
```

[14]:
```
# Plot training and validation loss
plt.figure(figsize=(12, 6))
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend()
plt.title('Training and Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.show()
```

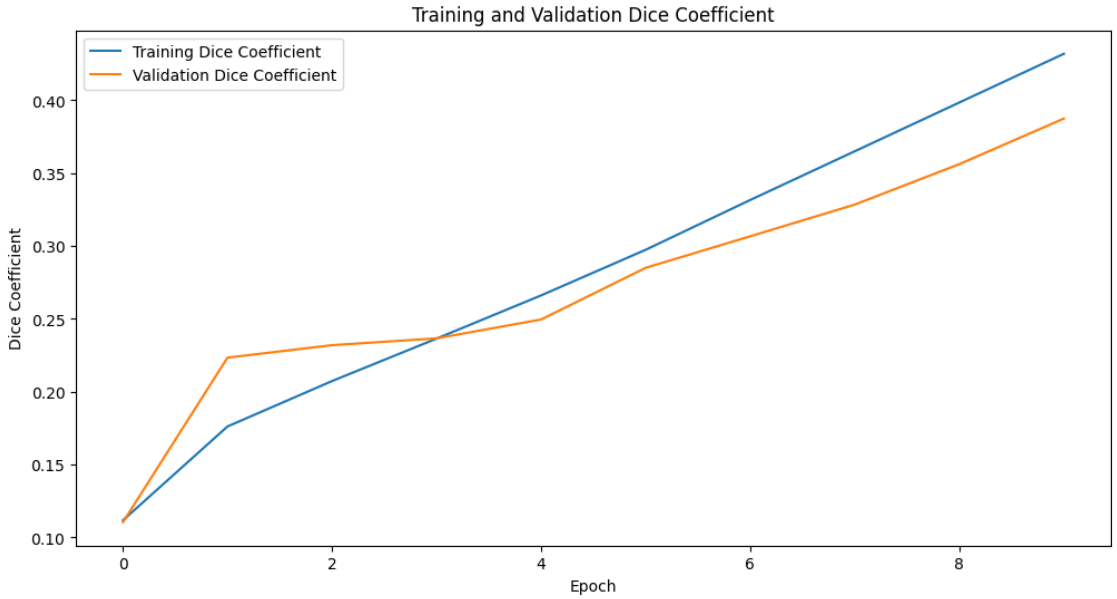


[15]:
```
# Plot training and validation accuracy
plt.figure(figsize=(12, 6))
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.title('Training and Validation Accuracy')
```

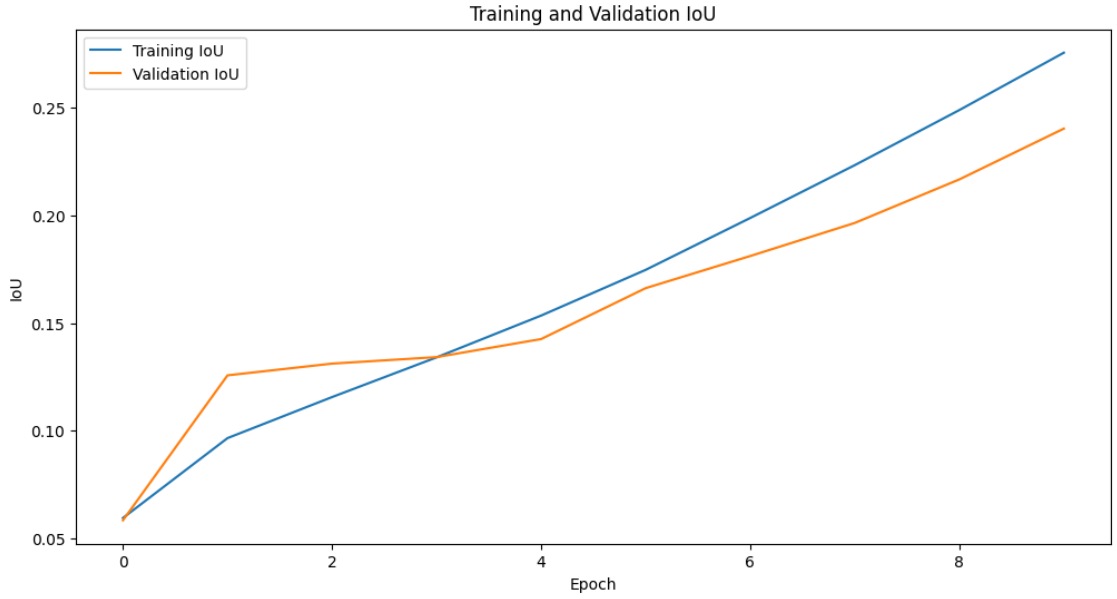
```
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.show()
```



[16]:
```
# Plot training and validation Dice coefficient
plt.figure(figsize=(12, 6))
plt.plot(history.history['dice_coefficient'], label='Training Dice Coefficient')
plt.plot(history.history['val_dice_coefficient'], label='Validation Dice␣
↪Coefficient')
plt.legend()
plt.title('Training and Validation Dice Coefficient')
plt.xlabel('Epoch')
plt.ylabel('Dice Coefficient')
plt.show()
```

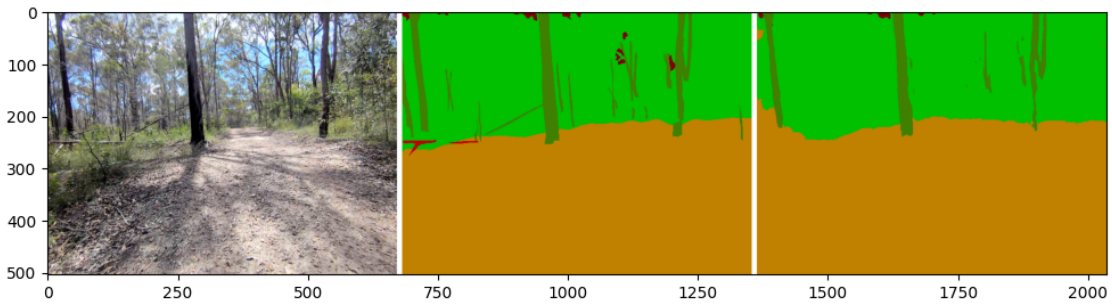


[17]:
```
# Plot training and validation IoU
plt.figure(figsize=(12, 6))
plt.plot(history.history['iou'], label='Training IoU')
plt.plot(history.history['val_iou'], label='Validation IoU')
plt.legend()
plt.title('Training and Validation IoU')
plt.xlabel('Epoch')
plt.ylabel('IoU')
plt.show()
```

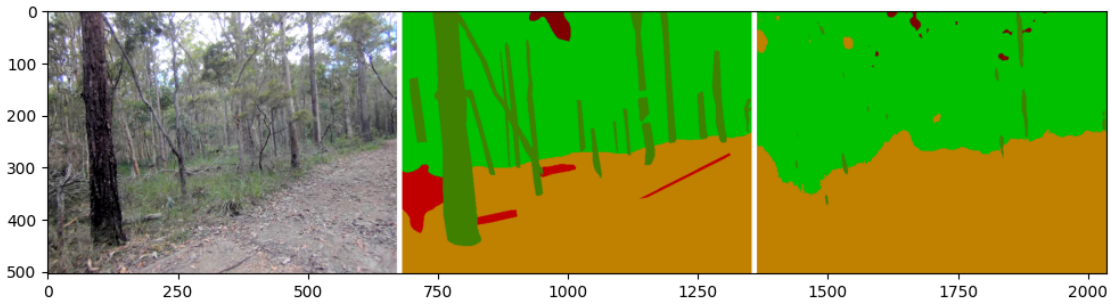


[18]:
```
# Visualize prediction results
for i, (x_path, y_path) in enumerate(zip(test_x[:10], test_y[:10])):
    x = read_image(x_path)
    y_pred = model.predict(np.expand_dims(x, axis=0))[0]
    y_original = read_mask(y_path)  # Directly read the mask

    h, w, _ = x.shape
    white_line = np.ones((h, 10, 3)) * 255

    all_images = [
        x * 255, white_line,
        mask_parse(y_original), white_line,  # Parse and display the original␣
↪mask
        mask_parse(y_pred)
    ]
    image = np.concatenate(all_images, axis=1).astype(np.uint8)

    fig = plt.figure(figsize=(12, 12))
    a = fig.add_subplot(1, 1, 1)
    imgplot = plt.imshow(image)
    plt.show()
```

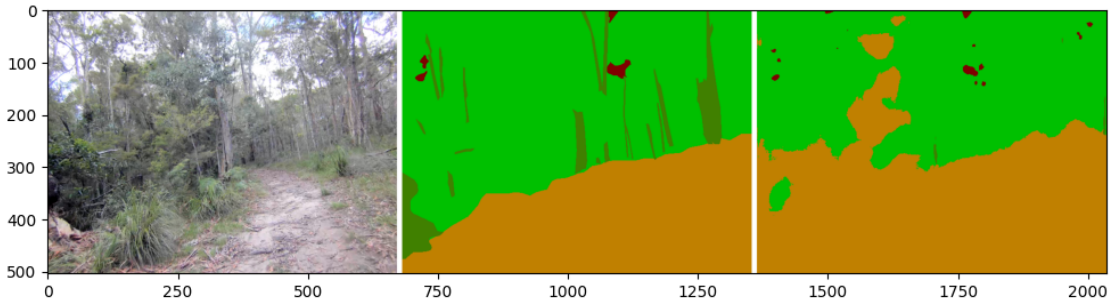
```
1/1             2s 2s/step
```



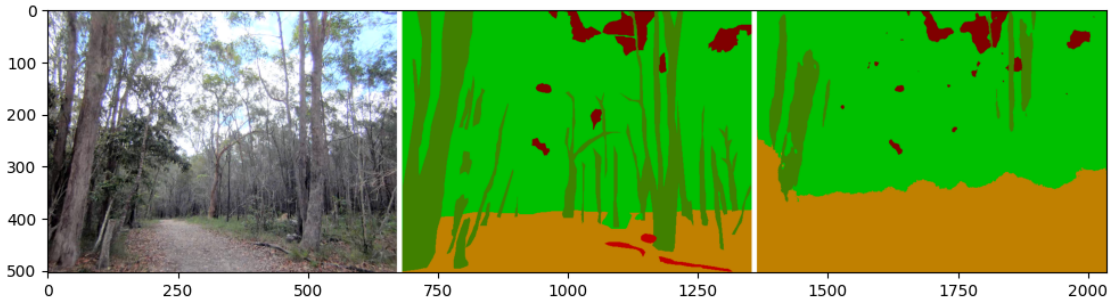

```
1/1             0s 11ms/step
```



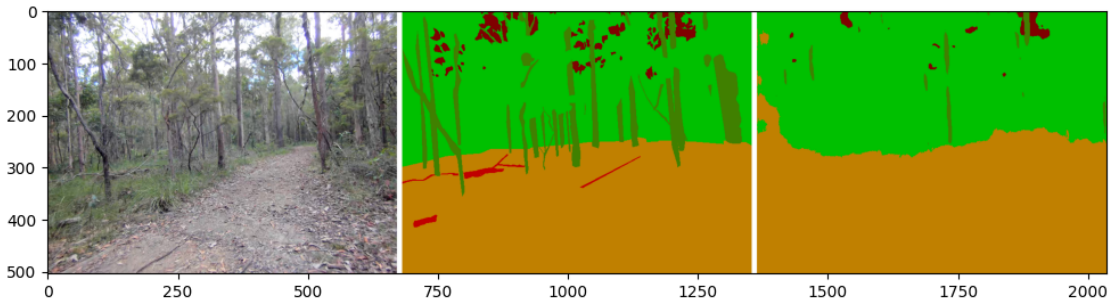

```
1/1             0s 12ms/step
```



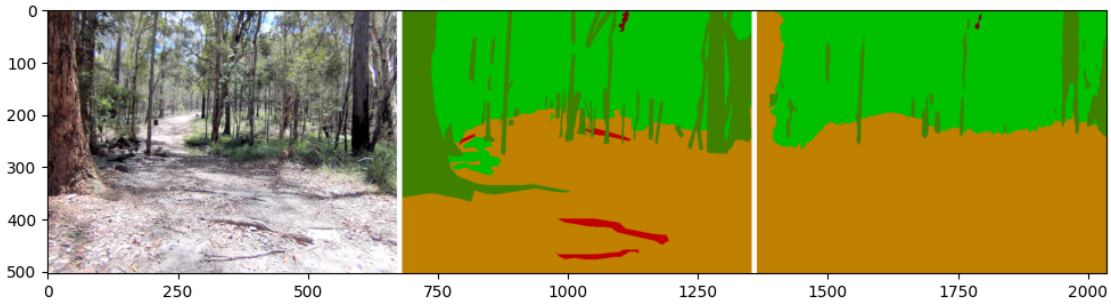

```
1/1             0s 11ms/step
```



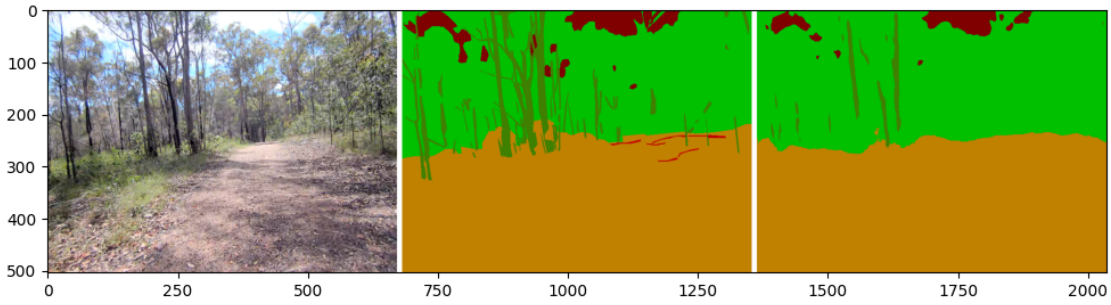

```
1/1             0s 11ms/step
```



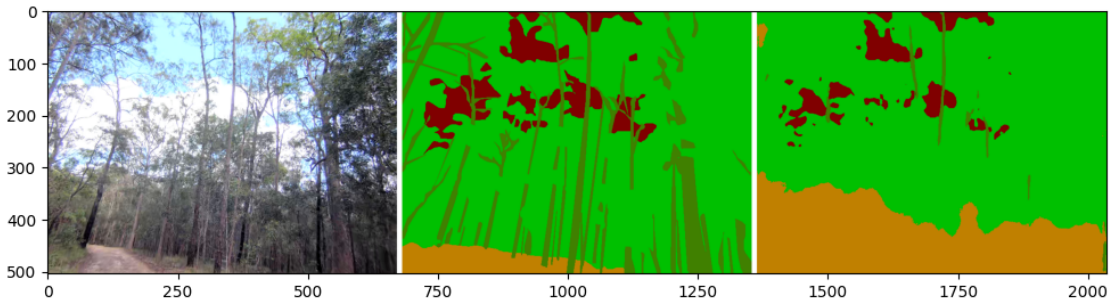

```
1/1             0s 11ms/step
```



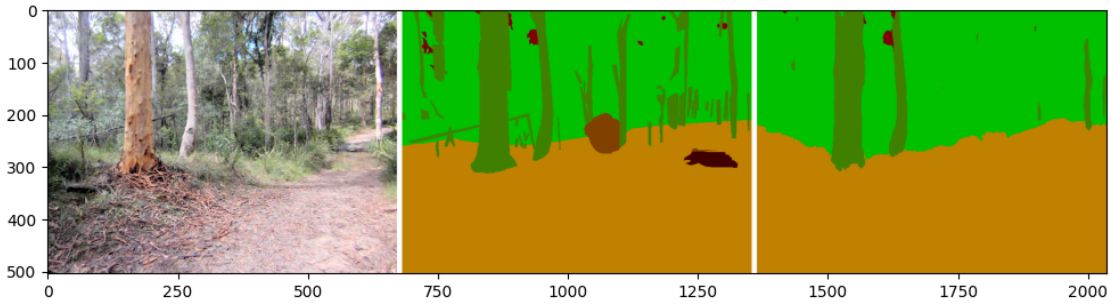

```
1/1             0s 10ms/step
```




```
1/1             0s 11ms/step
```




```
1/1                0s 11ms/step
```




```
1/1                0s 11ms/step
```