Untitled6

November 18, 2023

[31]:
```
import tensorflow as tf
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.mobilenet_v2 import MobileNetV2,␣
 ↪preprocess_input, decode_predictions
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D
```

[32]:
```
# Define the number of classes in your dataset
num_classes = 2  # Change this according to the number of classes in your␣
 ↪dataset

# Load MobileNetV2 base model pre-trained on ImageNet data
base_model = MobileNetV2(weights='imagenet', include_top=False)

# Freeze the layers of the base model
for layer in base_model.layers:
    layer.trainable = False

# Create a custom model on top of the base model
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation='relu')(x)
x = tf.keras.layers.Dropout(0.5)(x)
predictions = Dense(num_classes, activation='softmax')(x)

model = Model(inputs=base_model.input, outputs=predictions)

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy',␣
 ↪metrics=['accuracy'])
```

```
WARNING:tensorflow:`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.
```

[34]:
```
datagen = tf.keras.preprocessing.image.ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
```

```
    zoom_range=0.2,
    horizontal_flip=True,
    validation_split=0.2  # 20% of the data will be used for validation
)
```

[37]:
```
# Specify the path to your animal dataset
train_data_dir = 'cats_and_dogs_filtered/cats_and_dogs_filtered/train'
validation_data_dir = 'cats_and_dogs_filtered/cats_and_dogs_filtered/validation'

# Generate training dataset
train_generator = datagen.flow_from_directory(
    train_data_dir,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='training',
   )

# Generate validation dataset
validation_generator = datagen.flow_from_directory(
    validation_data_dir,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='validation',
   )
```

```
Found 1568 images belonging to 2 classes.
Found 200 images belonging to 2 classes.
```

[38]:
```
# Train the model
history = model.fit(
    train_generator,
    epochs=10,
    validation_data=validation_generator
)
```

```
Epoch 1/10
49/49 [==============================] - 82s 2s/step - loss: 0.0954 - accuracy:
0.9694 - val_loss: 0.0604 - val_accuracy: 0.9800
Epoch 2/10
49/49 [==============================] - 81s 2s/step - loss: 0.0846 - accuracy:
0.9745 - val_loss: 0.0719 - val_accuracy: 0.9800
Epoch 3/10
49/49 [==============================] - 80s 2s/step - loss: 0.1106 - accuracy:
0.9700 - val_loss: 0.0922 - val_accuracy: 0.9650
Epoch 4/10
49/49 [==============================] - 82s 2s/step - loss: 0.0608 - accuracy:
```

```
0.9783 - val_loss: 0.0879 - val_accuracy: 0.9650
Epoch 5/10
49/49 [==============================] - 80s 2s/step - loss: 0.0639 - accuracy:
0.9809 - val_loss: 0.0485 - val_accuracy: 0.9850
Epoch 6/10
49/49 [==============================] - 87s 2s/step - loss: 0.0594 - accuracy:
0.9815 - val_loss: 0.1469 - val_accuracy: 0.9650
Epoch 7/10
49/49 [==============================] - 85s 2s/step - loss: 0.0365 - accuracy:
0.9904 - val_loss: 0.0908 - val_accuracy: 0.9650
Epoch 8/10
49/49 [==============================] - 90s 2s/step - loss: 0.0378 - accuracy:
0.9866 - val_loss: 0.0636 - val_accuracy: 0.9900
Epoch 9/10
49/49 [==============================] - 82s 2s/step - loss: 0.0773 - accuracy:
0.9745 - val_loss: 0.1312 - val_accuracy: 0.9600
Epoch 10/10
49/49 [==============================] - 85s 2s/step - loss: 0.0563 - accuracy:
0.9796 - val_loss: 0.1193 - val_accuracy: 0.9750
```

[52]:
```python
# Specify the path to your test dataset
test_data_dir = 'cats_and_dogs_filtered/cats_and_dogs_filtered/test'

# Use ImageDataGenerator for normalization
test_datagen = tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)

# Generate test dataset
test_generator = test_datagen.flow_from_directory(
    test_data_dir,
    target_size=(224, 224),
    batch_size=32,
    class_mode=None,
    shuffle=False
)

# Make predictions on the test set
predictions = model.predict(test_generator)

# Decode the true labels and predicted labels
true_labels = test_generator.classes
predicted_labels = np.argmax(predictions, axis=1)

print("True Labels:", true_labels)  # Print the true labels for the first 10␣
 ↪images
print("Predicted Labels:", predicted_labels)  # Print the predicted labels for␣
 ↪the first 10 images
```

```
# Display the images along with true and predicted labels for all images in the test dataset
for i in range(len(true_labels)):
    img_path = test_generator.filepaths[i]
    img = image.load_img(img_path, target_size=(224, 224))
    img_array = image.img_to_array(img)

    # Display the image
    plt.imshow(img_array / 255.0)
    plt.axis('off')
    plt.show()

    # Display true and predicted labels
    true_label = 'cat' if true_labels[i] == 0 else 'dog'
    predicted_label = 'cat' if predicted_labels[i] == 0 else 'dog'

    print(f"Image {i+1}: True Label - {true_label}, Predicted Label - {predicted_label}")
```

```
Found 40 images belonging to 2 classes.
2/2 [==============================] - 1s 210ms/step
True Labels: [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1
 1 1 1]
Predicted Labels: [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1
 1 1 1]
```

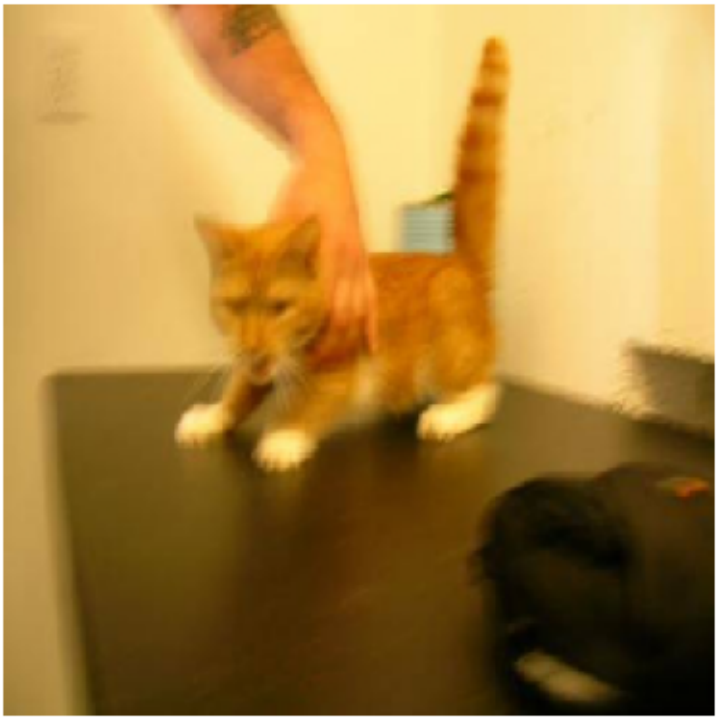

Image 1: True Label - cat, Predicted Label - cat

Image 2: True Label - cat, Predicted Label - cat

Image 3: True Label - cat, Predicted Label - cat

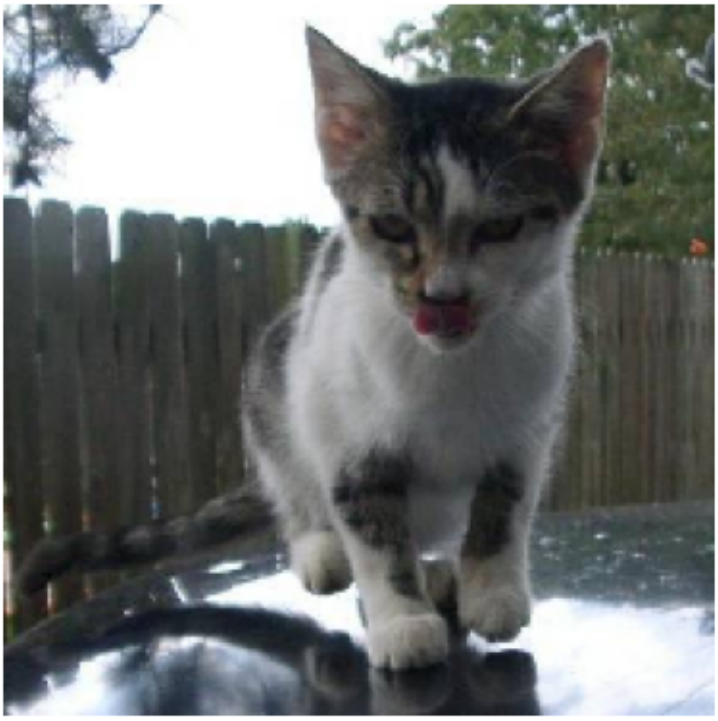

Image 4: True Label - cat, Predicted Label - cat

Image 5: True Label - cat, Predicted Label - cat

Image 6: True Label - cat, Predicted Label - cat

Image 7: True Label - cat, Predicted Label - cat

Image 8: True Label - cat, Predicted Label - cat

Image 9: True Label - cat, Predicted Label - cat

Image 10: True Label - cat, Predicted Label - cat

Image 11: True Label - cat, Predicted Label - cat

Image 12: True Label - cat, Predicted Label - cat

Image 13: True Label - cat, Predicted Label - cat

Image 14: True Label - cat, Predicted Label - cat

Image 15: True Label - cat, Predicted Label - cat

Image 16: True Label - cat, Predicted Label - cat

Image 17: True Label - cat, Predicted Label - dog

Image 18: True Label - cat, Predicted Label - cat

Image 19: True Label - cat, Predicted Label - cat

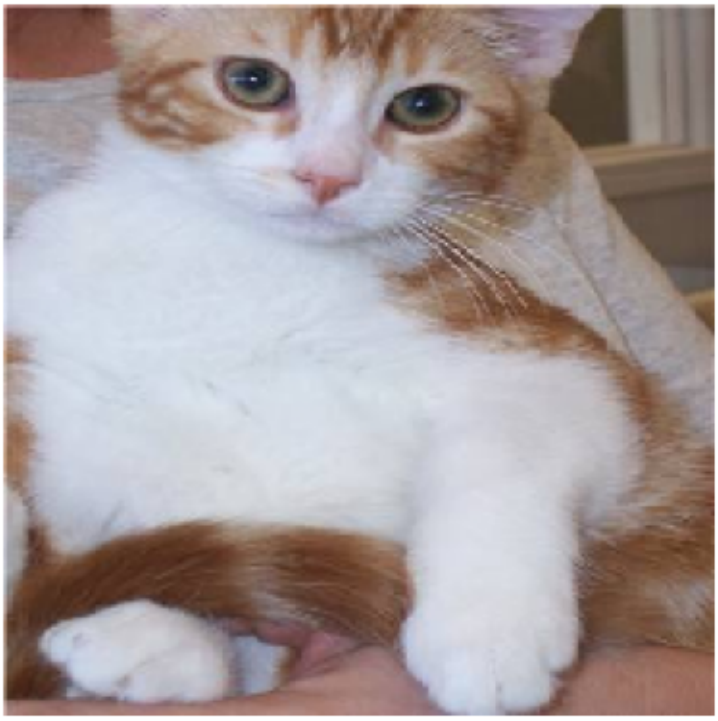

Image 20: True Label - cat, Predicted Label - cat

Image 21: True Label - dog, Predicted Label - dog

Image 22: True Label - dog, Predicted Label - dog

Image 23: True Label - dog, Predicted Label - dog

Image 24: True Label - dog, Predicted Label - dog

Image 25: True Label - dog, Predicted Label - dog

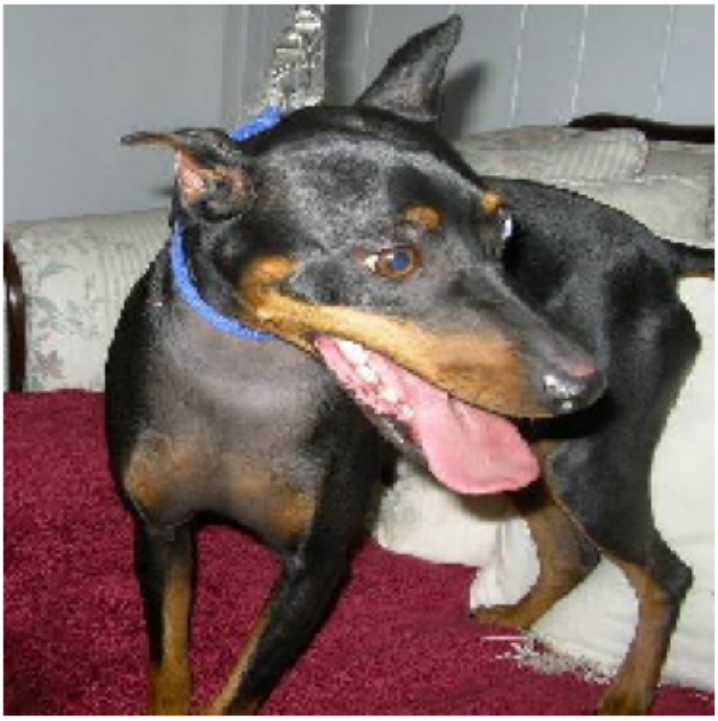

Image 26: True Label - dog, Predicted Label - dog

Image 27: True Label - dog, Predicted Label - dog

Image 28: True Label - dog, Predicted Label - dog

Image 29: True Label - dog, Predicted Label - dog

Image 30: True Label - dog, Predicted Label - dog

Image 31: True Label - dog, Predicted Label - dog

Image 32: True Label - dog, Predicted Label - dog

Image 33: True Label - dog, Predicted Label - dog

Image 34: True Label - dog, Predicted Label - dog

Image 35: True Label - dog, Predicted Label - dog

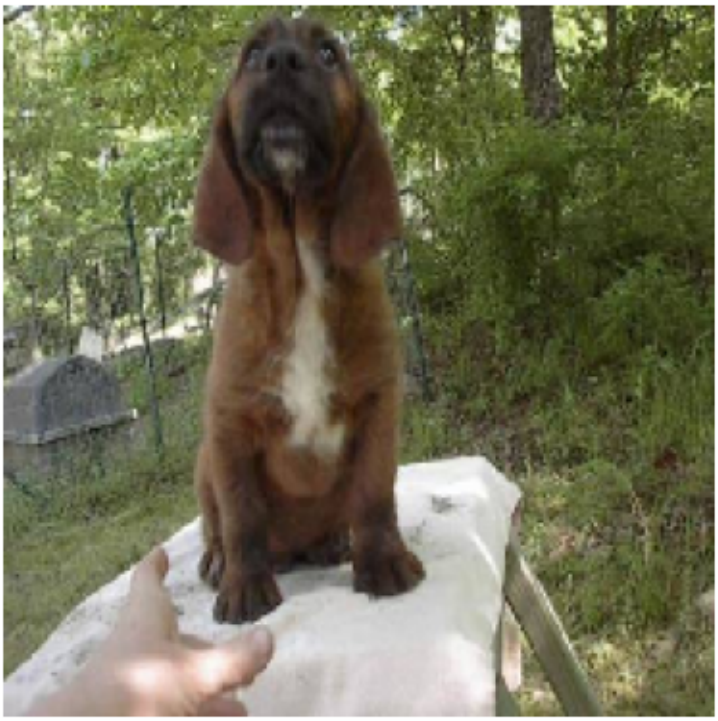

Image 36: True Label - dog, Predicted Label - dog

Image 37: True Label - dog, Predicted Label - dog

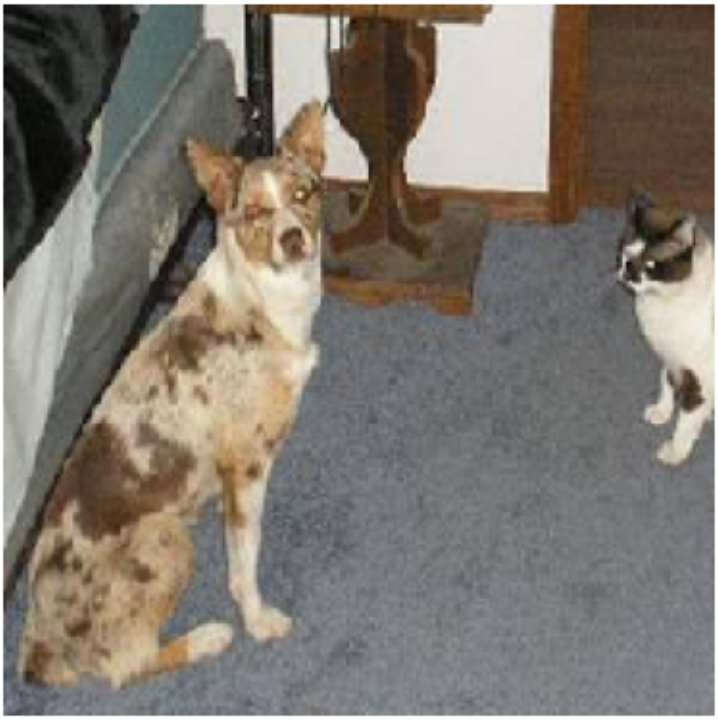

Image 38: True Label - dog, Predicted Label - dog

Image 39: True Label - dog, Predicted Label - dog

Image 40: True Label - dog, Predicted Label - dog

[ ]: