Image Classification with Intel Dataset

Authors

Muhammad Zaid, and Jiarui Zhao

The dataset I am using for this assignment is the [Intel Image Classification Dataset] {https://www.kaggle.com/datasets/puneet6060/intel-image-classification%7D found on kaggle. This dataset contains 25,000 images of 6 different classes: buildings, forest, glacier, mountain, sea, and street.

In the next few code blocks I will showcase the properties of the dataset, especially the number of images per class, the size of the images, and the number of images in the training and test sets.

Install All Necessary Packages

In [ ]:
```
%pip install "tensorflow-cpu"
```

```
Requirement already satisfied: tensorflow-cpu in c:\users\muzai\appdata\local\progra
ms\python\python311\lib\site-packages (2.12.0)
Requirement already satisfied: tensorflow-intel==2.12.0 in c:\users\muzai\appdata\lo
cal\programs\python\python311\lib\site-packages (from tensorflow-cpu) (2.12.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\muzai\appdata\local\progra
ms\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-cp
u) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\muzai\appdata\local\pro
grams\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-
cpu) (1.6.3)
Requirement already satisfied: flatbuffers>=2.0 in c:\users\muzai\appdata\local\prog
rams\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-c
pu) (23.3.3)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in c:\users\muzai\appdata\local\p
rograms\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflo
w-cpu) (0.4.0)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\muzai\appdata\local\p
rograms\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflo
w-cpu) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\muzai\appdata\local\programs
\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-cpu)
(3.8.0)
Requirement already satisfied: jax>=0.3.15 in c:\users\muzai\appdata\local\programs
\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-cpu)
(0.4.8)
Requirement already satisfied: libclang>=13.0.0 in c:\users\muzai\appdata\local\prog
rams\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-c
pu) (16.0.0)
Requirement already satisfied: numpy<1.24,>=1.22 in c:\users\muzai\appdata\local\pro
grams\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-
cpu) (1.23.5)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\muzai\appdata\local\pro
grams\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-
cpu) (3.3.0)
Requirement already satisfied: packaging in c:\users\muzai\appdata\roaming\python\py
thon311\site-packages (from tensorflow-intel==2.12.0->tensorflow-cpu) (23.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.
4,!=4.21.5,<5.0.0dev,>=3.20.3 in c:\users\muzai\appdata\local\programs\python\python
311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-cpu) (4.22.3)
Requirement already satisfied: setuptools in c:\users\muzai\appdata\local\programs\p
ython\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-cpu) (6
5.5.0)
Requirement already satisfied: six>=1.12.0 in c:\users\muzai\appdata\roaming\python
\python311\site-packages (from tensorflow-intel==2.12.0->tensorflow-cpu) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\muzai\appdata\local\prog
rams\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow-c
pu) (2.2.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\muzai\appdata\lo
cal\programs\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tens
orflow-cpu) (4.5.0)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in c:\users\muzai\appdata\local\p
rograms\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflo
w-cpu) (1.14.1)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\muzai\appdata\local\p
rograms\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflo
w-cpu) (1.54.0)
```

```
Requirement already satisfied: tensorboard<2.13,>=2.12 in c:\users\muzai\appdata\loc
al\programs\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tenso
rflow-cpu) (2.12.2)
Requirement already satisfied: tensorflow-estimator<2.13,>=2.12.0 in c:\users\muzai
\appdata\local\programs\python\python311\lib\site-packages (from tensorflow-intel==
2.12.0->tensorflow-cpu) (2.12.0)
Requirement already satisfied: keras<2.13,>=2.12.0 in c:\users\muzai\appdata\local\p
rograms\python\python311\lib\site-packages (from tensorflow-intel==2.12.0->tensorflo
w-cpu) (2.12.0)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in c:\users\muza
i\appdata\local\programs\python\python311\lib\site-packages (from tensorflow-intel==
2.12.0->tensorflow-cpu) (0.31.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\muzai\appdata\local\pr
ograms\python\python311\lib\site-packages (from astunparse>=1.6.0->tensorflow-intel=
=2.12.0->tensorflow-cpu) (0.40.0)
Requirement already satisfied: ml-dtypes>=0.0.3 in c:\users\muzai\appdata\local\prog
rams\python\python311\lib\site-packages (from jax>=0.3.15->tensorflow-intel==2.12.0-
>tensorflow-cpu) (0.1.0)
Requirement already satisfied: scipy>=1.7 in c:\users\muzai\appdata\local\programs\p
ython\python311\lib\site-packages (from jax>=0.3.15->tensorflow-intel==2.12.0->tenso
rflow-cpu) (1.10.1)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\muzai\appdata\local
\programs\python\python311\lib\site-packages (from tensorboard<2.13,>=2.12->tensorfl
ow-intel==2.12.0->tensorflow-cpu) (2.17.3)
Requirement already satisfied: google-auth-oauthlib<1.1,>=0.5 in c:\users\muzai\appd
ata\local\programs\python\python311\lib\site-packages (from tensorboard<2.13,>=2.12-
>tensorflow-intel==2.12.0->tensorflow-cpu) (1.0.0)
Requirement already satisfied: markdown>=2.6.8 in c:\users\muzai\appdata\local\progr
ams\python\python311\lib\site-packages (from tensorboard<2.13,>=2.12->tensorflow-int
el==2.12.0->tensorflow-cpu) (3.4.3)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\muzai\appdata\local\p
rograms\python\python311\lib\site-packages (from tensorboard<2.13,>=2.12->tensorflow
-intel==2.12.0->tensorflow-cpu) (2.28.2)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in c:\users\muz
ai\appdata\local\programs\python\python311\lib\site-packages (from tensorboard<2.13,
>=2.12->tensorflow-intel==2.12.0->tensorflow-cpu) (0.7.0)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in c:\users\muzai\appda
ta\local\programs\python\python311\lib\site-packages (from tensorboard<2.13,>=2.12->
tensorflow-intel==2.12.0->tensorflow-cpu) (1.8.1)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\muzai\appdata\local\progr
ams\python\python311\lib\site-packages (from tensorboard<2.13,>=2.12->tensorflow-int
el==2.12.0->tensorflow-cpu) (2.2.3)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\muzai\appdata\loca
l\programs\python\python311\lib\site-packages (from google-auth<3,>=1.6.3->tensorboa
rd<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow-cpu) (5.3.0)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\muzai\appdata\local
\programs\python\python311\lib\site-packages (from google-auth<3,>=1.6.3->tensorboar
d<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow-cpu) (0.3.0)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\muzai\appdata\local\program
s\python\python311\lib\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.13,>
=2.12->tensorflow-intel==2.12.0->tensorflow-cpu) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\muzai\appdata\lo
cal\programs\python\python311\lib\site-packages (from google-auth-oauthlib<1.1,>=0.5
->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow-cpu) (1.3.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\muzai\appdata\lo
cal\programs\python\python311\lib\site-packages (from requests<3,>=2.21.0->tensorboa
```

```
rd<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow-cpu) (3.1.0)
Requirement already satisfied: idna<4,>=2.5 in c:\users\muzai\appdata\local\programs
\python\python311\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.13,>=2.
12->tensorflow-intel==2.12.0->tensorflow-cpu) (3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\muzai\appdata\local
\programs\python\python311\lib\site-packages (from requests<3,>=2.21.0->tensorboard<
2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow-cpu) (1.26.15)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\muzai\appdata\local\pr
ograms\python\python311\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.1
3,>=2.12->tensorflow-intel==2.12.0->tensorflow-cpu) (2022.12.7)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\muzai\appdata\local\pro
grams\python\python311\lib\site-packages (from werkzeug>=1.0.1->tensorboard<2.13,>=
2.12->tensorflow-intel==2.12.0->tensorflow-cpu) (2.1.2)
Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in c:\users\muzai\appdata\local
\programs\python\python311\lib\site-packages (from pyasn1-modules>=0.2.1->google-aut
h<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow-cpu) (0.
5.0)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\muzai\appdata\local\progr
ams\python\python311\lib\site-packages (from requests-oauthlib>=0.7.0->google-auth-o
authlib<1.1,>=0.5->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow-cp
u) (3.2.2)
Note: you may need to restart the kernel to use updated packages.
[notice] A new release of pip is available: 23.0.1 -> 23.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

Data Loading And Exploration

In [ ]:

```
from PIL import Image

import numpy as np
import os

image = Image.open(os.getcwd()+'\\archive\\seg_train\\seg_train\\forest\\8.jpg')


print("Image Size of Images in Dataset: ", image.size)
print("Image Mode of Images in Dataset: ", image.mode)
imageCount:int = 0
classNum:list = []

for dirname, _, filenames in os.walk(os.getcwd()+'\\archive\\seg_train\\seg_train')
    for filename in filenames:
        imageCount += 1


    classNum.append((imageCount, dirname.split('\\')[-1]))
    imageCount = 0


classNum.pop(0)

print("Number of Images in Dataset: ", sum([i[0] for i in classNum]))
print("Number of Classes in Dataset: ", len(classNum))
print("Number of Images in each Class: ", classNum)
```

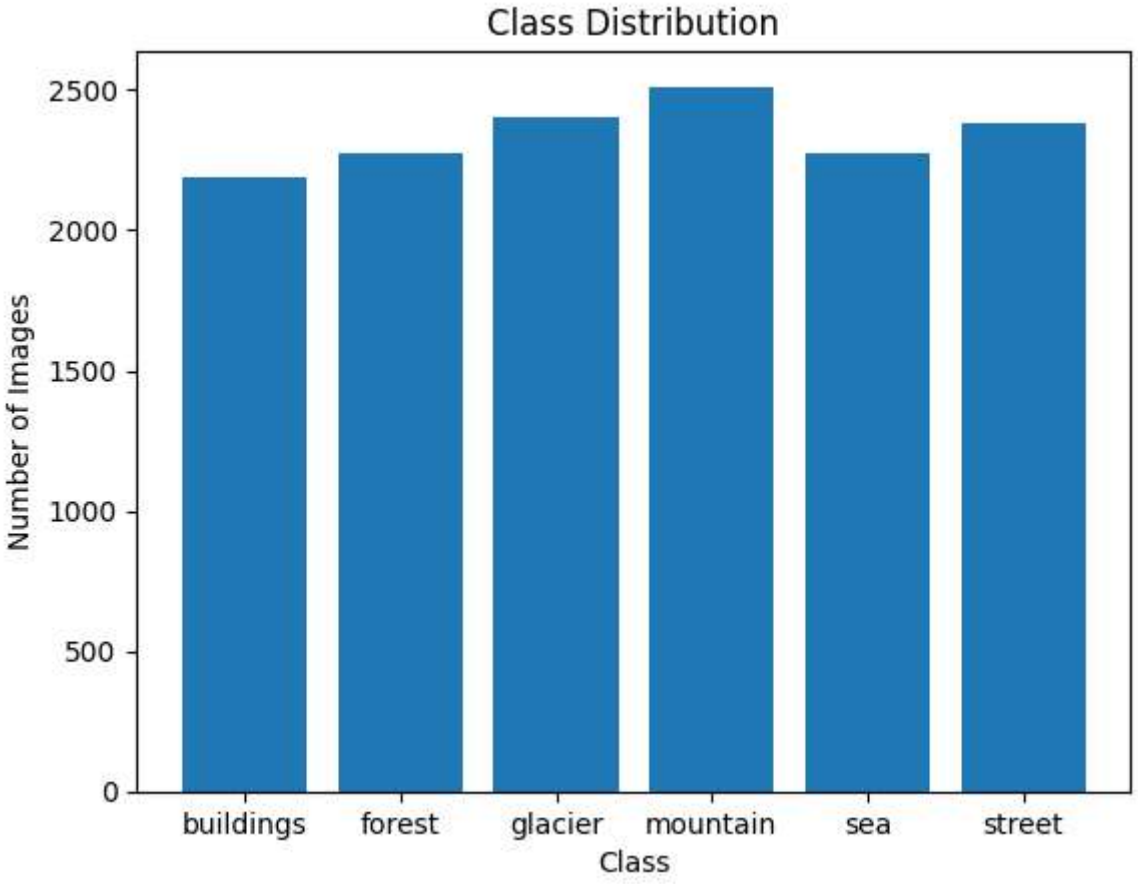
```
Image Size of Images in Dataset:  (150, 150)
Image Mode of Images in Dataset:  RGB
Number of Images in Dataset:  14034
Number of Classes in Dataset:  6
Number of Images in each Class:  [(2191, 'buildings'), (2271, 'forest'), (2404, 'gla
cier'), (2512, 'mountain'), (2274, 'sea'), (2382, 'street')]
```

As can be see in the ouput above the following is true:

- There are 6 classes of images
- Each class has an almosrt equal number of images
- The image size of each image is 150x150 pixels
- All images are in the RGB format

Now to display the distribution of the dataset

In [ ]:
```python
import matplotlib.pyplot as plt

# Extract the class names and values into separate lists
classes = [d[1] for d in classNum]
values = [d[0] for d in classNum]

# Create a bar chart of the values for each class
plt.bar(classes, values)

# Set the chart title and axis labels
plt.title('Class Distribution')
plt.xlabel('Class')
plt.ylabel('Number of Images')

# Show the chart
plt.show()
```



Since we have an equal distribution of images per class, we can hope for high recall and precision for each class.
The dataset with its six different classes can employ a multi class classification cnn model. So we would be predicting based on the images in the testing set which class the image belongs to.

The dataset is split into a training, testing, and prediction set. The prediction set does not have any class labels.

Lets show a image from each class, so that there is a visual representation of the classes.

In [ ]:
```
import glob

cwd = os.getcwd() + "\\archive\\seg_train\\seg_train\\"
subdirs = [f.path for f in os.scandir(cwd) if f.is_dir()]

listOfFiles:list = []

# Iterate over each subdirectory and find the first file
for subdir in subdirs:
    classname = os.path.basename(subdir)
    # Get a list of all files in the subdirectory
    files = glob.glob(subdir + '/*')
    if len(files) > 0:
        # Print the first file in the subdirectory
        listOfFiles.append((classname, files[0]))
```

```
plt.figure(figsize=(10,10))
for i in range(listOfFiles.__len__()):
    img = np.array(Image.open(listOfFiles[i][1]))
    plt.subplot(3,3,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(img)
    plt.xlabel(listOfFiles[i][0])
plt.show()
```


buildings


forest


glacier


mountain


sea


street

Now I need to preprocess the image to convert all images into nuympy arrays and then normalize the images. This way I can directoly feed the images into the model.

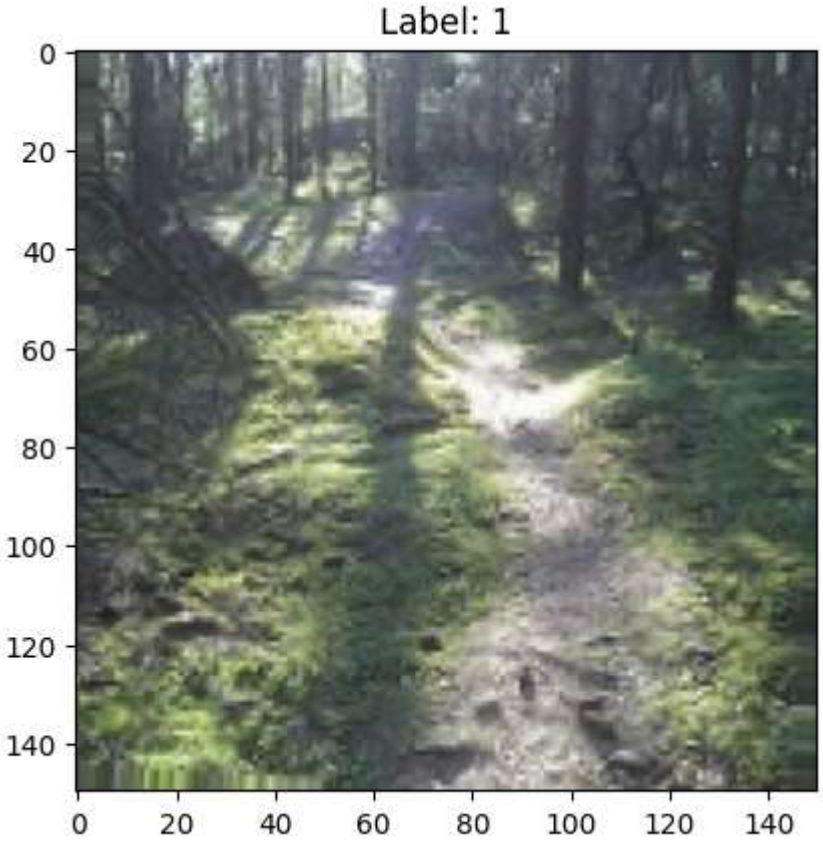
I also need to make sure that the dataset has a 80 to 20 split for training and testing.

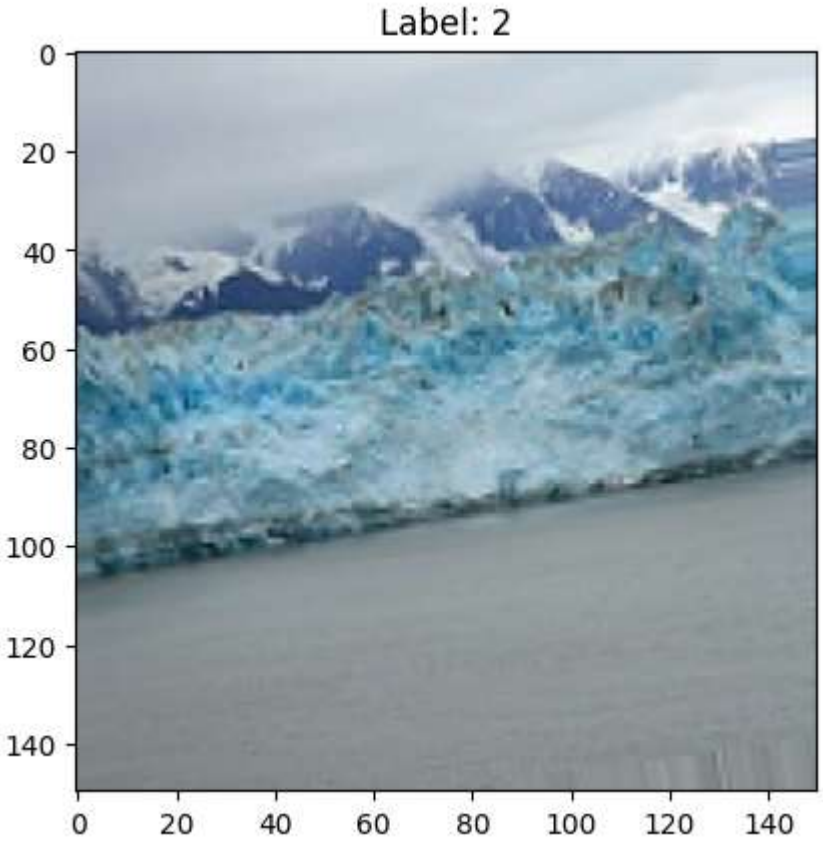
I will evaluate the results for two different models with and without color in the model.

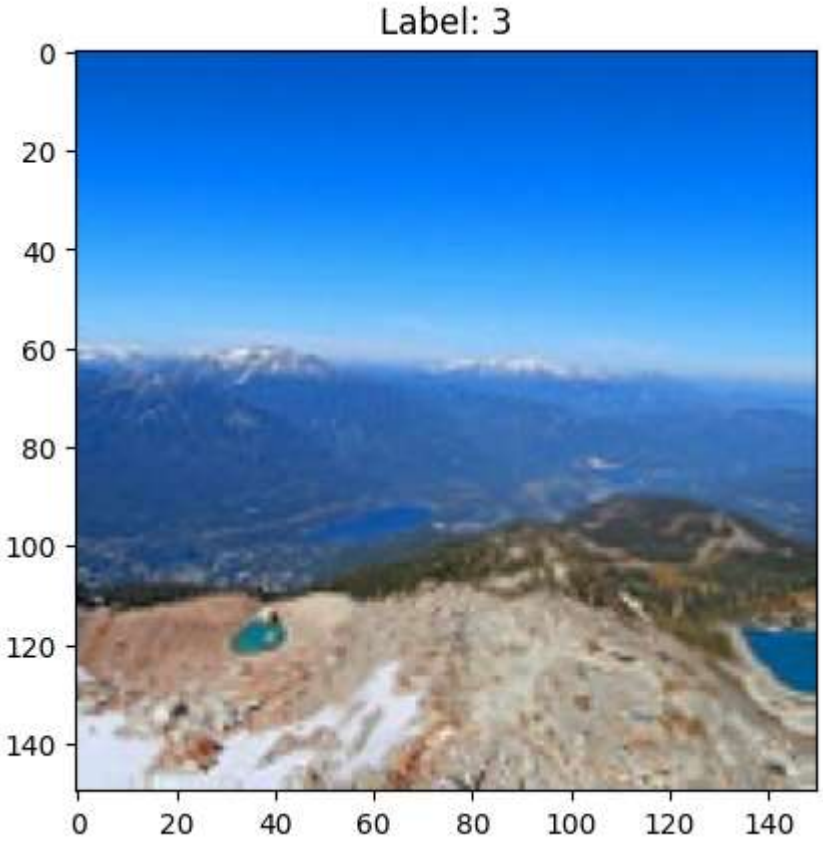
In [ ]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_url = os.getcwd() + "\\archive\\seg_train\\seg_train\\"
test_url = os.getcwd() + "\\archive\\seg_test\\seg_test\\"


datagen = ImageDataGenerator(
    rescale=1./255,  # rescale pixel values to be between 0 and 1
    rotation_range=10,  # randomly rotate images by up to 20 degrees
    horizontal_flip=True)  # randomly flip images horizontally
```

```
train = datagen.flow_from_directory(train_url, target_size=(150, 150), batch_size=2
test = datagen.flow_from_directory(test_url, target_size=(150, 150), batch_size=256


images, labels = next(train)

# Display the properties of the first few images
for i in range(3):
    plt.imshow(images[i])
    plt.title('Label: {}'.format(np.argmax(labels[i])))
    plt.show()
```

```
Found 14034 images belonging to 6 classes.
Found 3000 images belonging to 6 classes.
```







The Labels for each image indicate a number from 0 to 5, representing a class.
This datagen is for images with color.

And as can be seen by the number of images for each train and test, the split is roughly around 80 to 20.

Model Building

First, I will build a CNN based model for color images.
The model will have 4 convolutional layers, 4 max pooling layers, a flatten layer, 2 dense layers, and a droupout layer.

Second, I will build a CNN based model for grayscaler images.

Third, I will build a CNN based model for color images with a different architecture.
The model will have 3 convolutional layers, 3 max pooling layers, a flatten layer, 2 dense layers, and a droupout layer.
I want to see how decreasing the convolutional layers affects the model.

Fourth, I will use transfer learning to train a pretrained model on my dataset, could use RestNet or Yolo.

I will then evaluate all the results for the models, to see which of these performed the best.

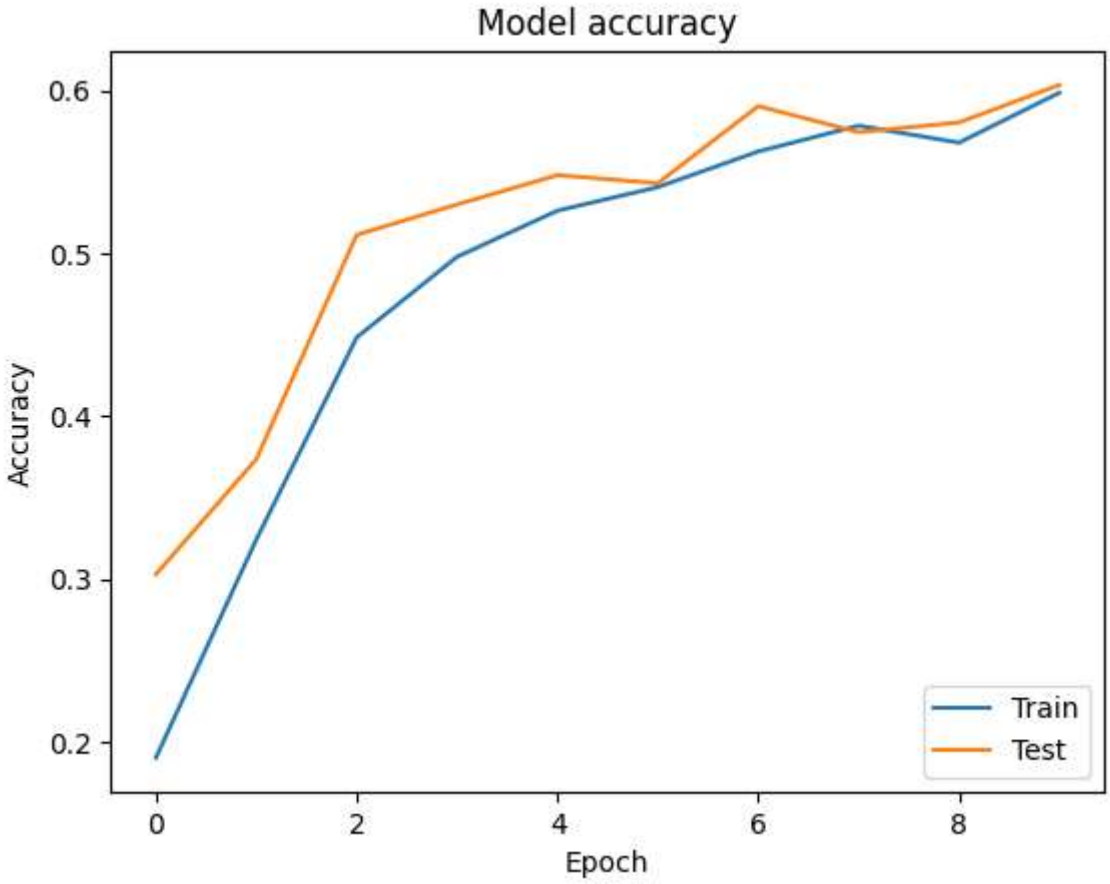
In [ ]:
```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150, 3)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(256, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(6, activation='softmax'))

model.summary()
```

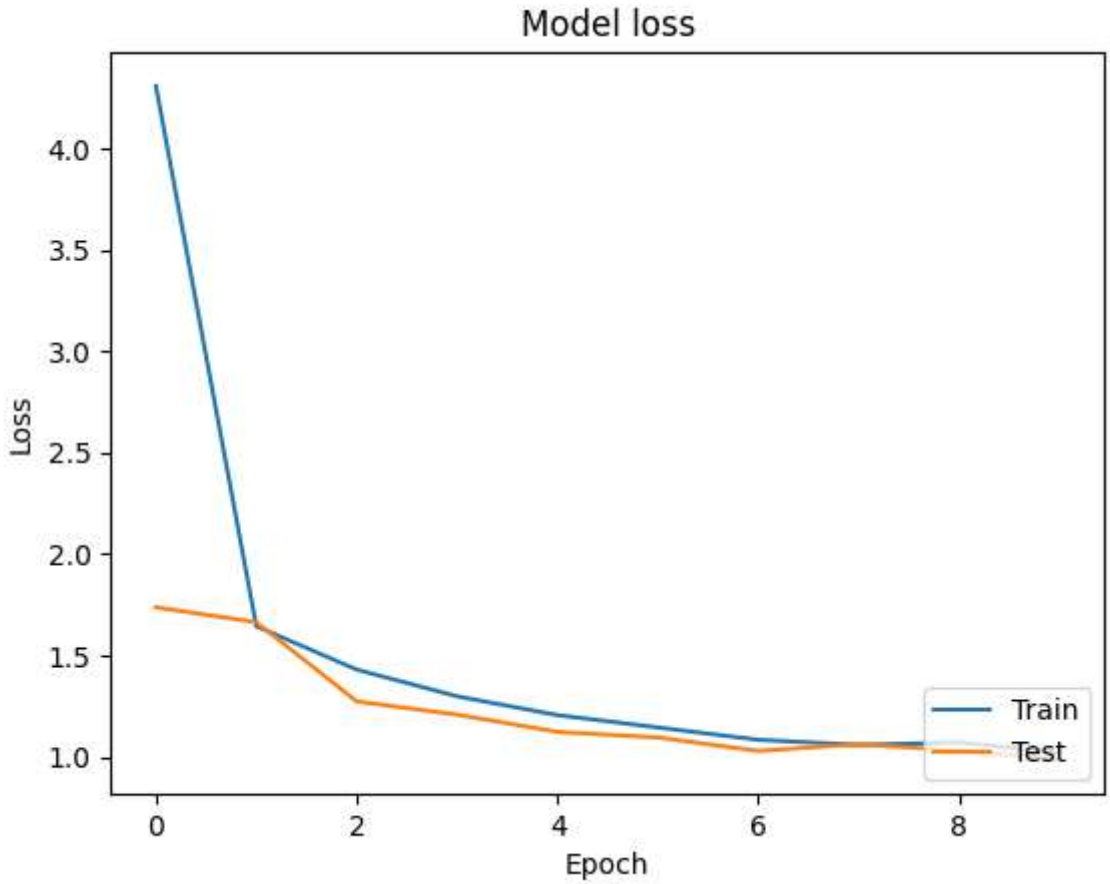
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 148, 148, 32)      896

 max_pooling2d (MaxPooling2D  (None, 74, 74, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 72, 72, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 36, 36, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 34, 34, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 17, 17, 128)      0
 2D)

 conv2d_3 (Conv2D)           (None, 15, 15, 128)       147584

 max_pooling2d_3 (MaxPooling  (None, 7, 7, 128)        0
 2D)

 conv2d_4 (Conv2D)           (None, 5, 5, 256)         295168

 max_pooling2d_4 (MaxPooling  (None, 2, 2, 256)        0
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 148, 148, 32)      896

 max_pooling2d (MaxPooling2D  (None, 74, 74, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 72, 72, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 36, 36, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 34, 34, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 17, 17, 128)      0
 2D)

 conv2d_3 (Conv2D)           (None, 15, 15, 128)       147584

 max_pooling2d_3 (MaxPooling  (None, 7, 7, 128)        0
 2D)

 conv2d_4 (Conv2D)           (None, 5, 5, 256)         295168

 max_pooling2d_4 (MaxPooling  (None, 2, 2, 256)        0
 2D)

 flatten (Flatten)           (None, 1024)              0
```

```
 dense (Dense)               (None, 512)               524800

 dropout (Dropout)           (None, 512)               0

 dense_1 (Dense)             (None, 6)                 3078

=================================================================
Total params: 1,063,878
Trainable params: 1,063,878
Non-trainable params: 0
_________________________________________________________________
```

In [ ]:
```
from tensorflow import keras

optimizer = keras.optimizers.Adam(learning_rate=0.001)

if os.path.exists('models/image_classification_model_2.h5'):
    model = keras.models.load_model('models/image_classification_model_1.h5')
else:
    model.compile(optimizer=optimizer, loss='categorical_crossentropy', metrics=['a

    fitted = model.fit(train, steps_per_epoch = 50 ,batch_size=128, epochs=10, vali
```

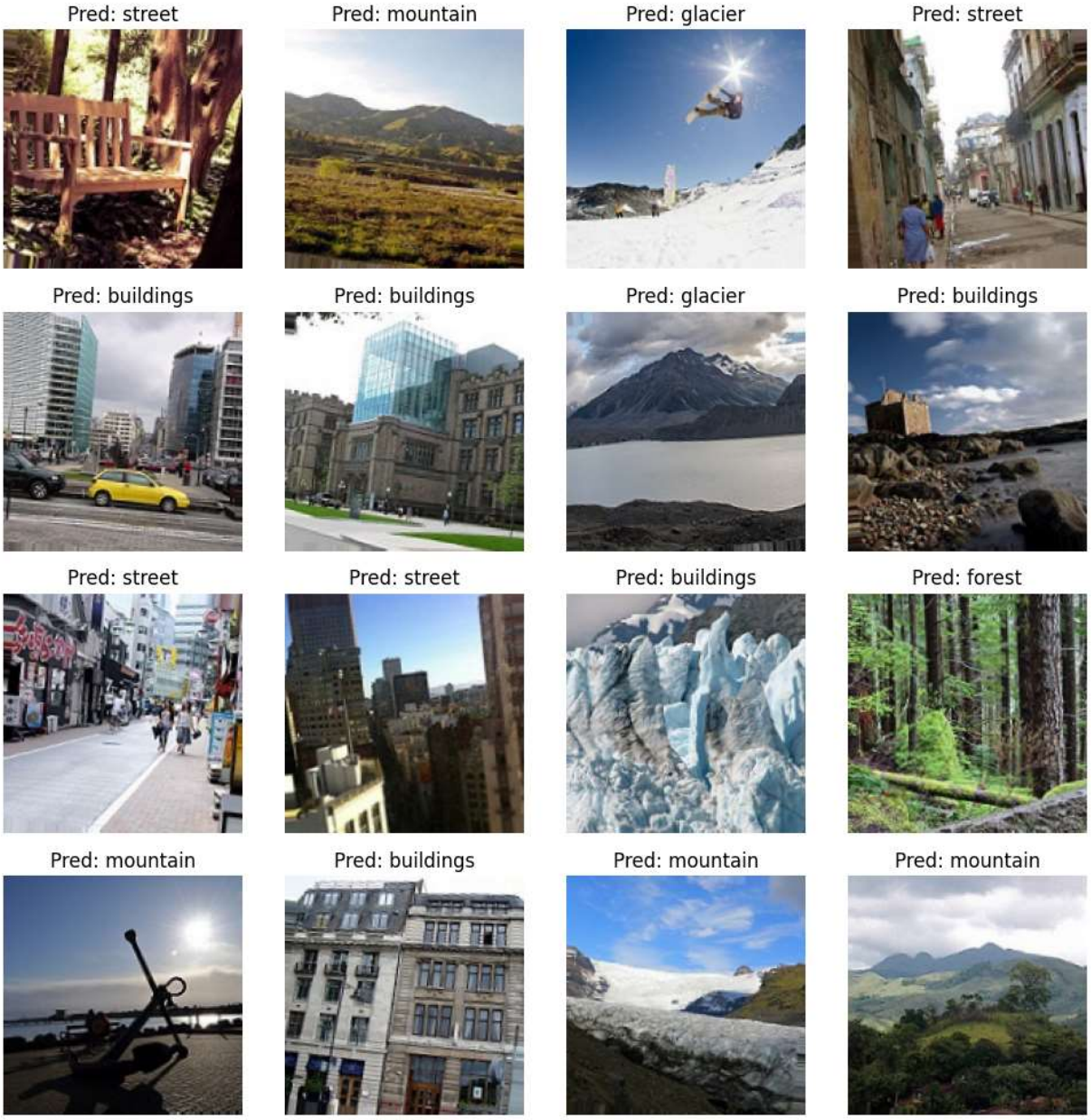
In [ ]:
```
import random
from tensorflow.keras.preprocessing import image

model.save('models/image_classification_model_1.h5')

plt.plot(fitted.history['accuracy'])
plt.plot(fitted.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='lower right')
plt.show()

plt.plot(fitted.history['loss'])
plt.plot(fitted.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='lower right')
plt.show()

cwd = os.getcwd() + "\\archive\\seg_pred\\seg_pred\\"
img_names = os.listdir(cwd)
random_img_names = random.sample(img_names, 5)

class_dict = {0: 'buildings', 1: 'forest', 2: 'glacier', 3: 'mountain', 4: 'sea', 5
```


Model accuracy
Accuracy
Epoch
Train
Test
0.2
0.3
0.4
0.5
0.6
0
2
4
6
8


Model loss
Loss
Epoch
Train
Test
1.0
1.5
2.0
2.5
3.0
3.5
4.0
0
2
4
6
8

In [ ]:
```
pred_url = os.getcwd() + "\\archive\\seg_pred"


datagen = ImageDataGenerator(
    rescale=1./255,  # rescale pixel values to be between 0 and 1
    rotation_range=10,  # randomly rotate images by up to 20 degrees
    horizontal_flip=True)  # randomly flip images horizontally

pred = datagen.flow_from_directory(pred_url, target_size=(150, 150), batch_size=256


import matplotlib.pyplot as plt
import numpy as np

# get the class labels
class_labels = list(train.class_indices.keys())

# get a batch of images and their predictions
x, _ = next(pred)
y_pred = model.predict(x)

# display the images and their predictions
fig, axs = plt.subplots(4, 4, figsize=(10, 10))
for i, ax in enumerate(axs.flatten()):
    # show the image
    ax.imshow(x[i])
    ax.axis('off')

    # show the predicted label
    pred_label = class_labels[np.argmax(y_pred[i])]
    ax.set_title(f'Pred: {pred_label}')

plt.tight_layout()
plt.show()
```

```
Found 7301 images belonging to 1 classes.
8/8 [==============================] - 1s 71ms/step
```



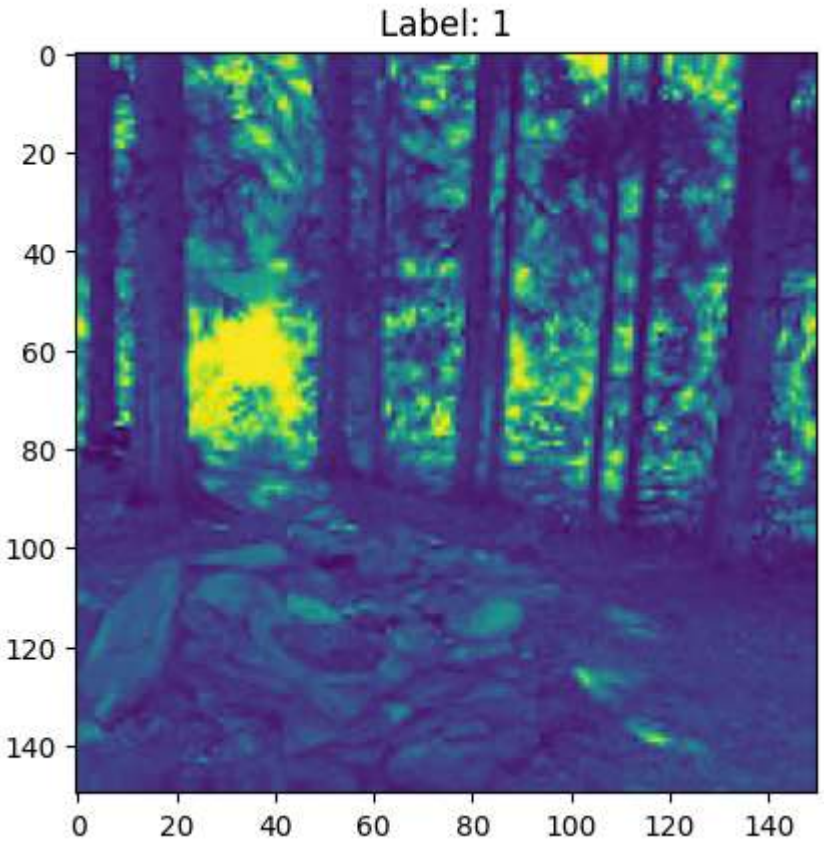
First model done with the color images.

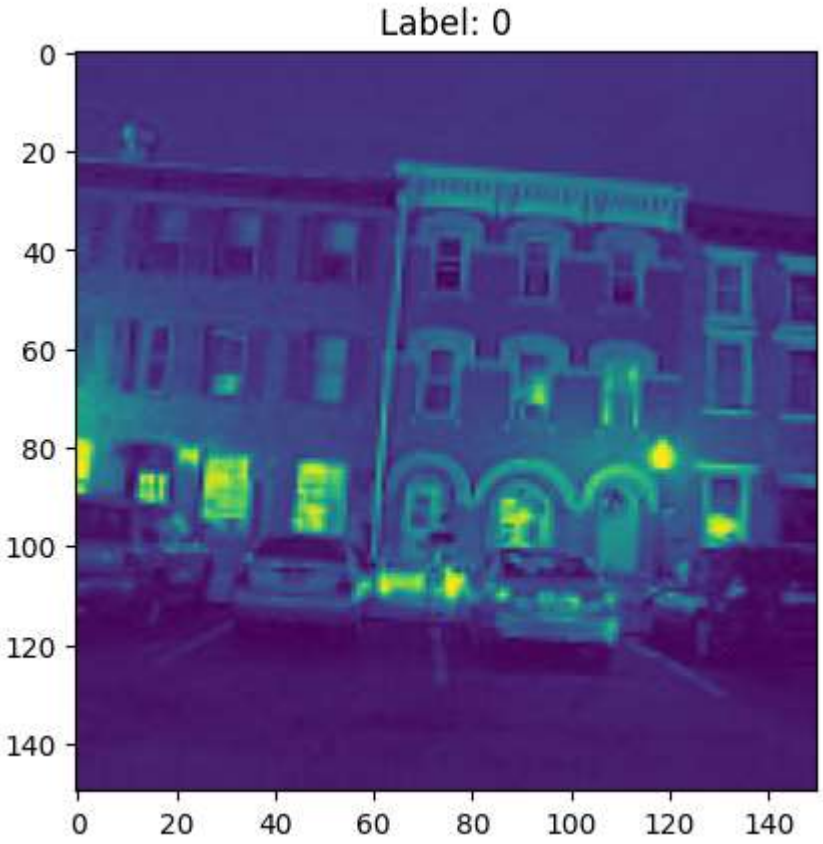
Now to train the same model with the grayscale images.
And then to train the same model with the color images but with a different architecture.
And then finally transfer learning.

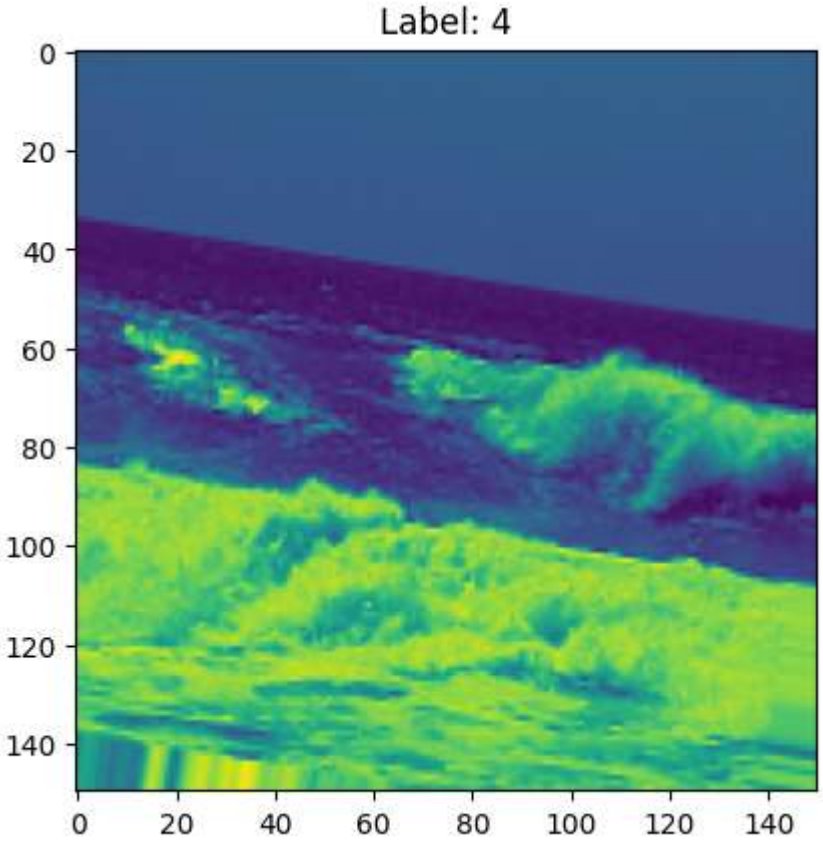
In [ ]:
```
train_url = os.getcwd() + "\\archive\\seg_train\\seg_train\\"
test_url = os.getcwd() + "\\archive\\seg_test\\seg_test\\"


datagen = ImageDataGenerator(
    rescale=1./255,  # rescale pixel values to be between 0 and 1
    rotation_range=10,  # randomly rotate images by up to 20 degrees
    horizontal_flip=True)  # randomly flip images horizontally

train_gs = datagen.flow_from_directory(train_url, target_size=(150, 150), batch_siz
test_gs = datagen.flow_from_directory(test_url, target_size=(150, 150), batch_size=
```

```
images, labels = next(train_gs)

# Display the properties of the first few images
for i in range(3):
    plt.imshow(images[i])
    plt.title('Label: {}'.format(np.argmax(labels[i])))
    plt.show()
```

```
Found 14034 images belonging to 6 classes.
Found 3000 images belonging to 6 classes.
```







In [ ]:
```
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150, 1)))
model.add(MaxPooling2D((2, 2)))
```

```
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(256, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(6, activation='softmax'))

model.summary()
```

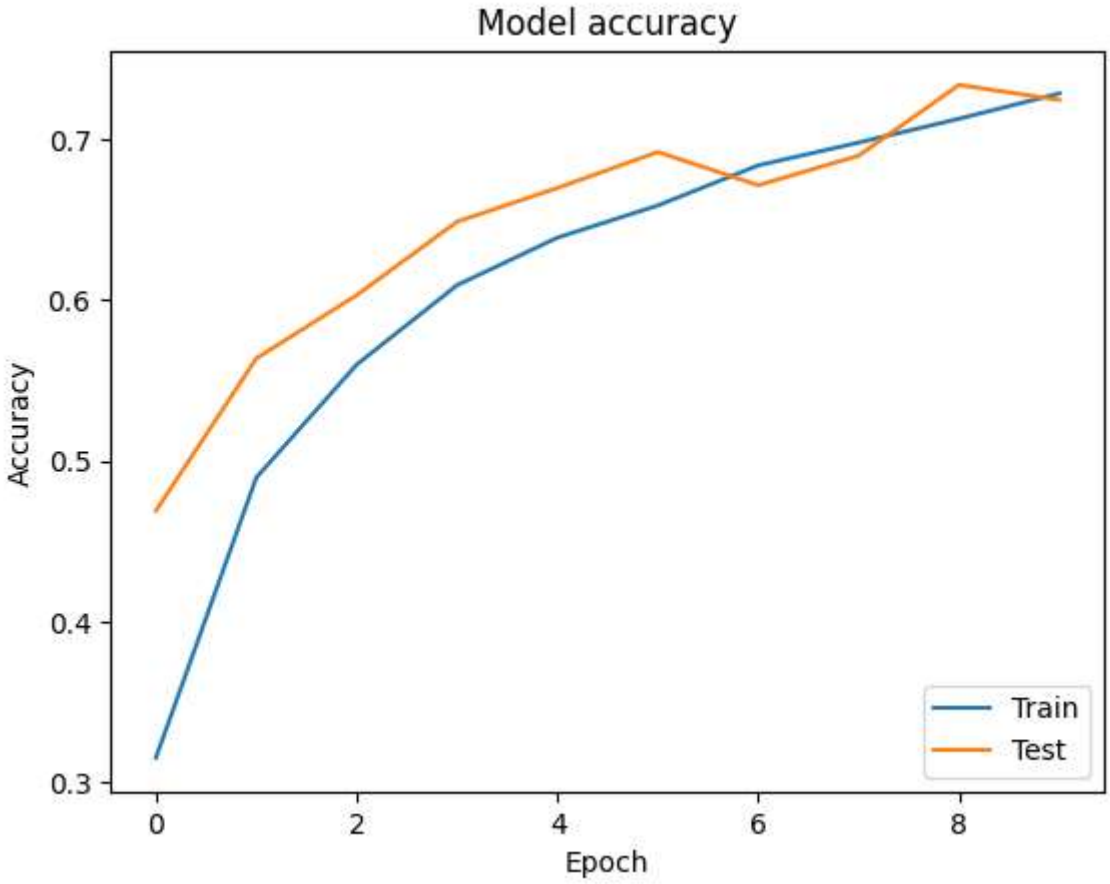
```
Model: "sequential_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_44 (Conv2D)          (None, 148, 148, 32)      320

 max_pooling2d_44 (MaxPoolin  (None, 74, 74, 32)       0
 g2D)

 conv2d_45 (Conv2D)          (None, 72, 72, 64)        18496

 max_pooling2d_45 (MaxPoolin  (None, 36, 36, 64)       0
 g2D)

 conv2d_46 (Conv2D)          (None, 34, 34, 128)       73856

 max_pooling2d_46 (MaxPoolin  (None, 17, 17, 128)      0
 g2D)

 conv2d_47 (Conv2D)          (None, 15, 15, 128)       147584

 max_pooling2d_47 (MaxPoolin  (None, 7, 7, 128)        0
 g2D)

 conv2d_48 (Conv2D)          (None, 5, 5, 256)         295168

 max_pooling2d_48 (MaxPoolin  (None, 2, 2, 256)        0
 g2D)

 flatten_9 (Flatten)         (None, 1024)              0

 dense_28 (Dense)            (None, 512)               524800

 dropout_9 (Dropout)         (None, 512)               0

 dense_29 (Dense)            (None, 6)                 3078

=================================================================
Total params: 1,063,302
Trainable params: 1,063,302
Non-trainable params: 0
_________________________________________________________________
```

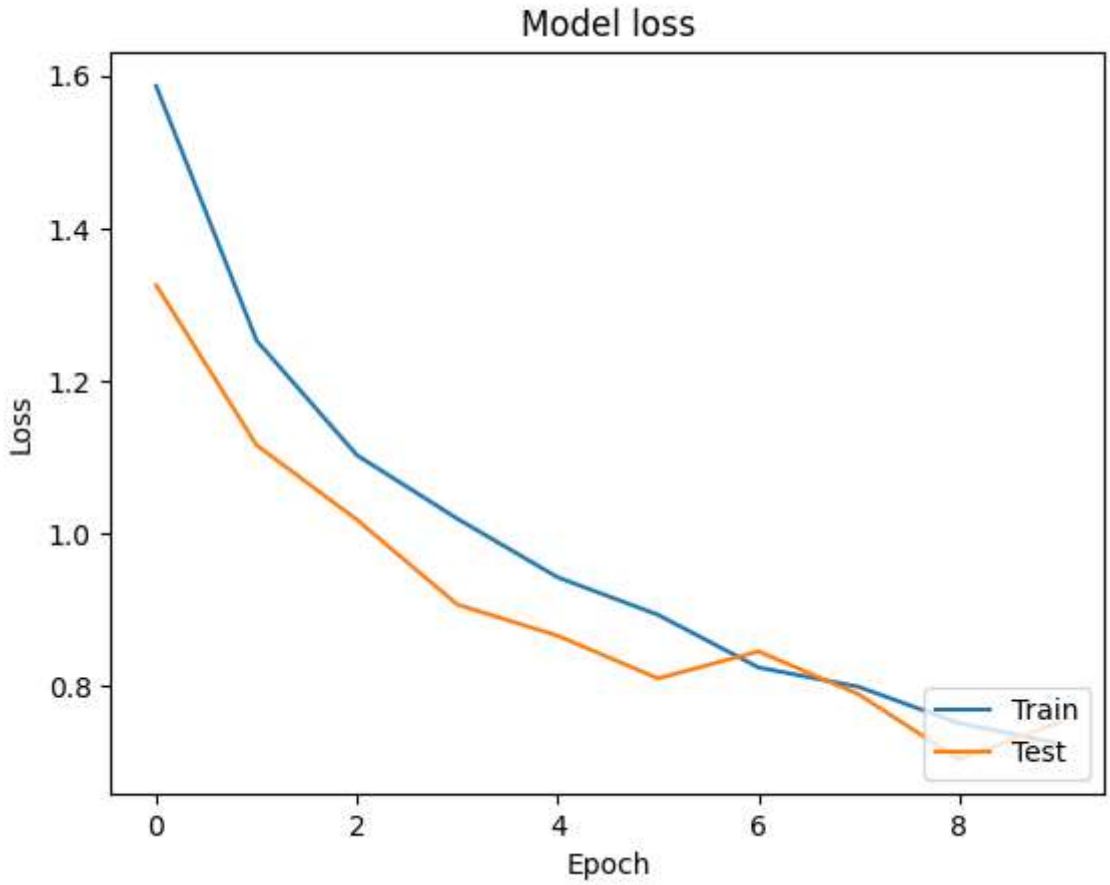
In [ ]:
```
from tensorflow import keras

optimizer = keras.optimizers.Adam(learning_rate=0.001)

if os.path.exists('models/image_classification_model_2.h5'):
    model = keras.load_model('models/image_classification_model_2.h5')
else:
    model.compile(optimizer=optimizer, loss='categorical_crossentropy', metrics=['a
    cnn_gray = model.fit(train_gs, steps_per_epoch=50, batch_size=128, epochs=10, v
    model.save('models/image_classification_model_2.h5')
```

```
Epoch 1/10
50/50 [==============================] - 73s 1s/step - loss: 1.5865 - accuracy: 0.31
55 - val_loss: 1.3255 - val_accuracy: 0.4690
Epoch 2/10
50/50 [==============================] - 70s 1s/step - loss: 1.2530 - accuracy: 0.48
96 - val_loss: 1.1159 - val_accuracy: 0.5637
Epoch 3/10
50/50 [==============================] - 73s 1s/step - loss: 1.1027 - accuracy: 0.55
98 - val_loss: 1.0181 - val_accuracy: 0.6030
Epoch 4/10
50/50 [==============================] - 74s 1s/step - loss: 1.0196 - accuracy: 0.60
92 - val_loss: 0.9071 - val_accuracy: 0.6487
Epoch 5/10
50/50 [==============================] - 72s 1s/step - loss: 0.9427 - accuracy: 0.63
88 - val_loss: 0.8662 - val_accuracy: 0.6697
Epoch 6/10
50/50 [==============================] - 72s 1s/step - loss: 0.8936 - accuracy: 0.65
87 - val_loss: 0.8102 - val_accuracy: 0.6920
Epoch 7/10
50/50 [==============================] - 72s 1s/step - loss: 0.8247 - accuracy: 0.68
37 - val_loss: 0.8457 - val_accuracy: 0.6713
Epoch 8/10
50/50 [==============================] - 72s 1s/step - loss: 0.7991 - accuracy: 0.69
78 - val_loss: 0.7893 - val_accuracy: 0.6897
Epoch 9/10
50/50 [==============================] - 73s 1s/step - loss: 0.7515 - accuracy: 0.71
27 - val_loss: 0.7040 - val_accuracy: 0.7337
Epoch 10/10
50/50 [==============================] - 72s 1s/step - loss: 0.7249 - accuracy: 0.72
84 - val_loss: 0.7532 - val_accuracy: 0.7243
```

In [ ]:

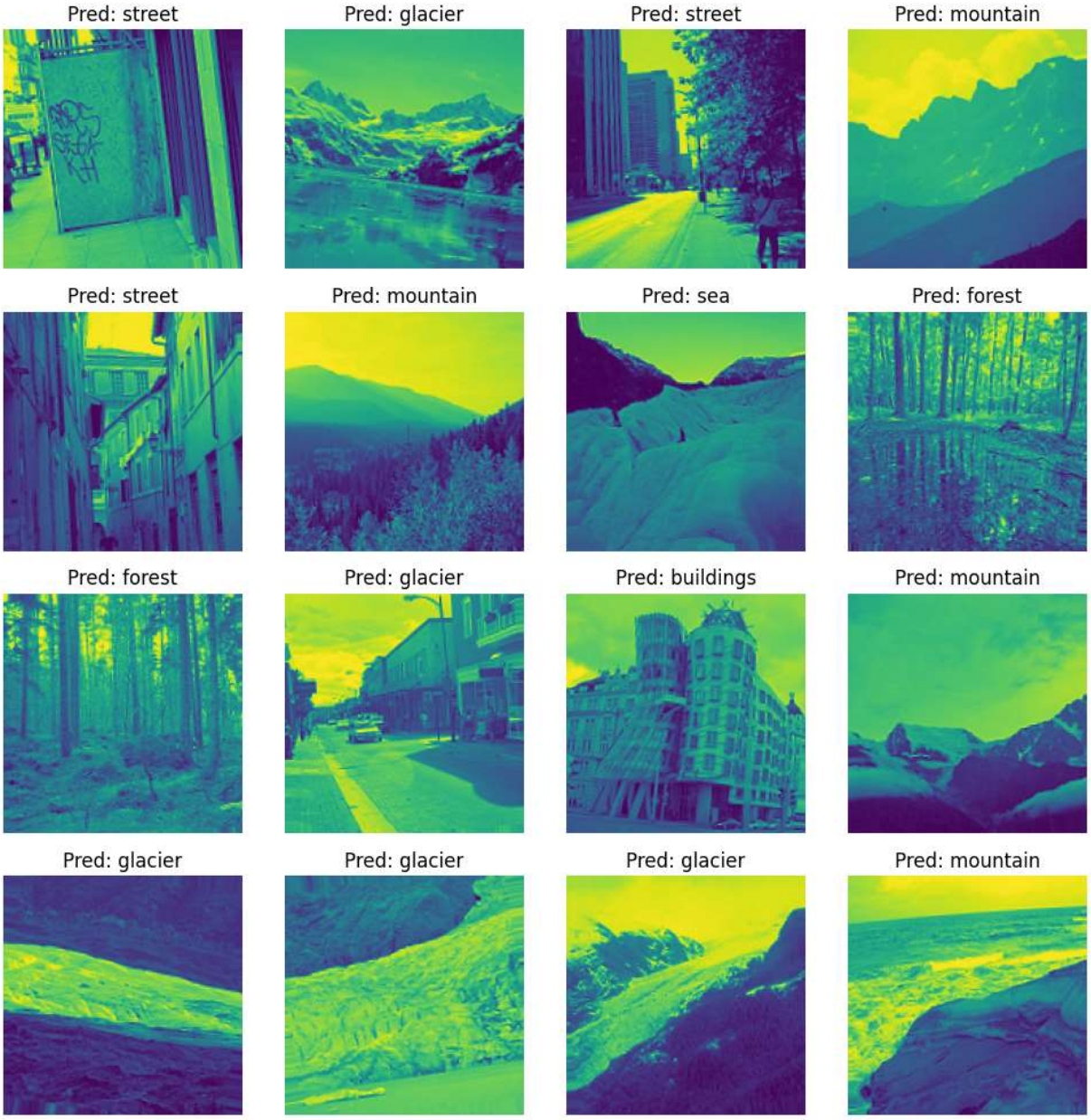
```
plt.plot(cnn_gray.history['accuracy'])
plt.plot(cnn_gray.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='lower right')
plt.show()

plt.plot(cnn_gray.history['loss'])
plt.plot(cnn_gray.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='lower right')
plt.show()

cwd = os.getcwd() + "\\archive\\seg_pred\\seg_pred\\"
img_names = os.listdir(cwd)
random_img_names = random.sample(img_names, 5)

class_dict = {0: 'buildings', 1: 'forest', 2: 'glacier', 3: 'mountain', 4: 'sea', 5
```


Model accuracy
Accuracy
Epoch
Train
Test
0.3
0.4
0.5
0.6
0.7
0
2
4
6
8


Model loss
Loss
Epoch
Train
Test
0.8
1.0
1.2
1.4
1.6
0
2
4
6
8

In [ ]:
```
pred_url = os.getcwd() + "\\archive\\seg_pred"


datagen = ImageDataGenerator(
    rescale=1./255,  # rescale pixel values to be between 0 and 1
    rotation_range=10,  # randomly rotate images by up to 20 degrees
    horizontal_flip=True)  # randomly flip images horizontally

pred = datagen.flow_from_directory(pred_url, target_size=(150, 150), batch_size=256


import matplotlib.pyplot as plt
import numpy as np

# get the class labels
class_labels = list(train.class_indices.keys())

# get a batch of images and their predictions
x, _ = next(pred)
y_pred = model.predict(x)

# display the images and their predictions
fig, axs = plt.subplots(4, 4, figsize=(10, 10))
for i, ax in enumerate(axs.flatten()):
    # show the image
    ax.imshow(x[i])
    ax.axis('off')

    # show the predicted label
    pred_label = class_labels[np.argmax(y_pred[i])]
    ax.set_title(f'Pred: {pred_label}')

plt.tight_layout()
plt.show()
```

```
Found 7301 images belonging to 1 classes.
8/8 [==============================] - 1s 74ms/step
```



Now to train with a different architecture.

In [ ]:
```
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150, 3)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(6, activation='softmax'))

model.summary()
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_40 (Conv2D)          (None, 148, 148, 32)      896

 max_pooling2d_40 (MaxPoolin  (None, 74, 74, 32)       0
 g2D)

 conv2d_41 (Conv2D)          (None, 72, 72, 64)        18496

 max_pooling2d_41 (MaxPoolin  (None, 36, 36, 64)       0
 g2D)

 conv2d_42 (Conv2D)          (None, 34, 34, 128)       73856

 max_pooling2d_42 (MaxPoolin  (None, 17, 17, 128)      0
 g2D)

 conv2d_43 (Conv2D)          (None, 15, 15, 128)       147584

 max_pooling2d_43 (MaxPoolin  (None, 7, 7, 128)        0
 g2D)

 flatten_8 (Flatten)         (None, 6272)              0

 dense_16 (Dense)            (None, 512)               3211776

 dropout_8 (Dropout)         (None, 512)               0

 dense_17 (Dense)            (None, 6)                 3078

=================================================================
Total params: 3,455,686
Trainable params: 3,455,686
Non-trainable params: 0
_________________________________________________________________
```

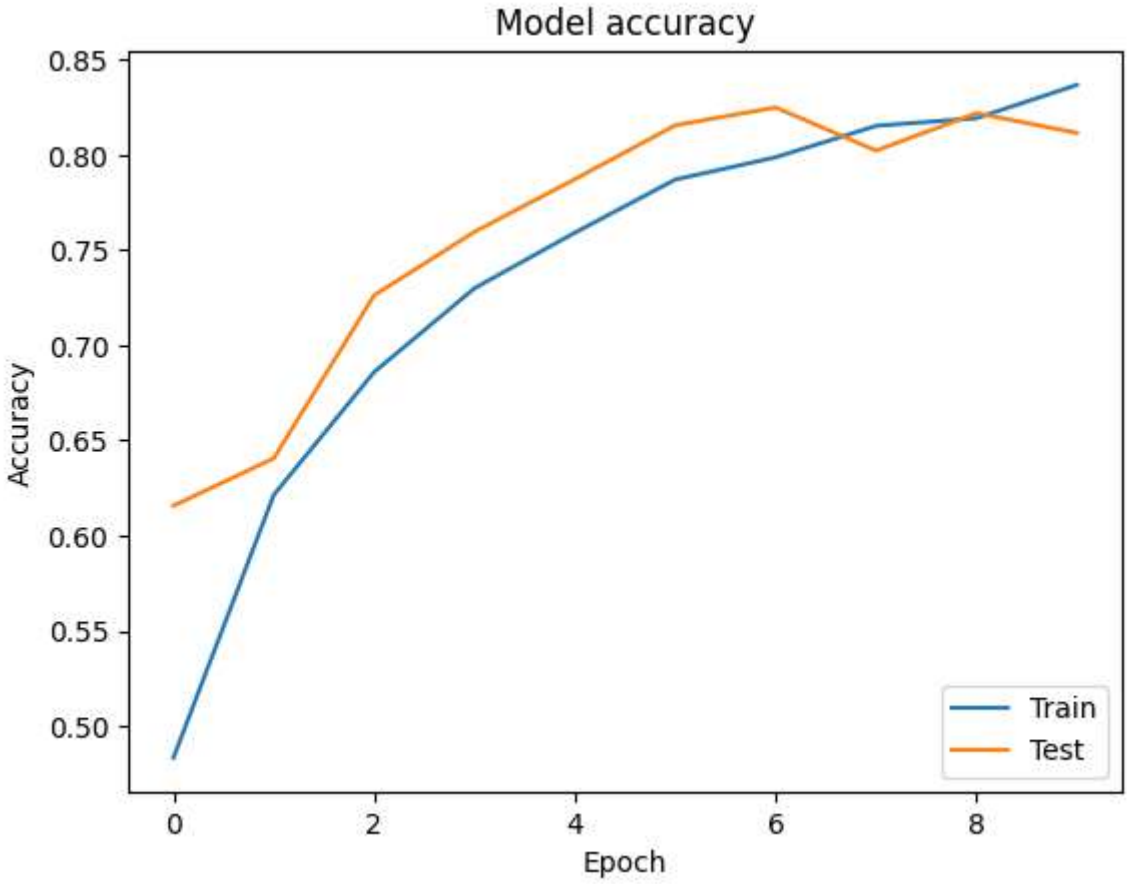
In [ ]:
```
from tensorflow import keras

optimizer = keras.optimizers.Adam(learning_rate=0.001)

if os.path.exists('models/image_classification_model_3.h5'):
    model = keras.load_model('models/image_classification_model_3.h5')

else:
    model.compile(optimizer=optimizer, loss='categorical_crossentropy', metrics=['a
    cnn = model.fit_generator(generator=train, steps_per_epoch=50, epochs=10, valid
    model.save('models/image_classification_model_3.h5')
```

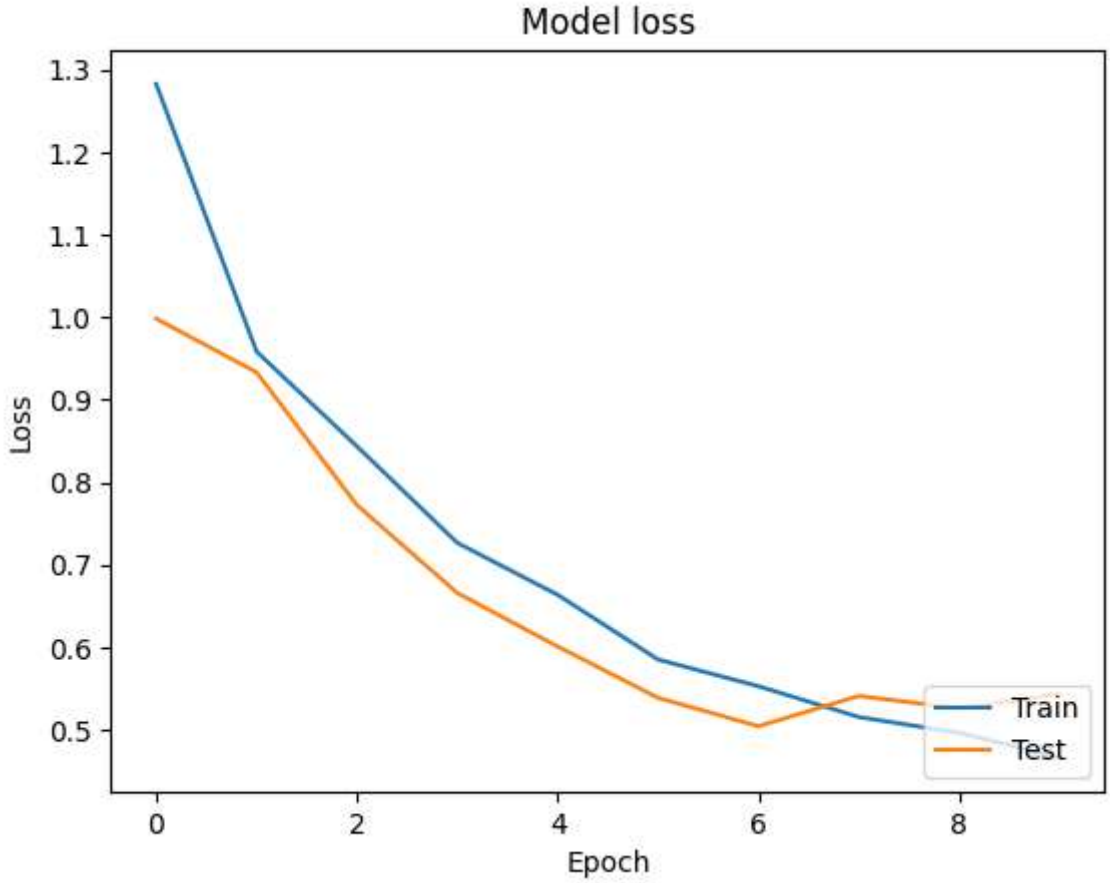
```
C:\Users\muzai\AppData\Local\Temp\ipykernel_11832\884340771.py:7: UserWarning: `Mode
l.fit_generator` is deprecated and will be removed in a future version. Please use `
Model.fit`, which supports generators.
  cnn = model.fit_generator(generator=train, steps_per_epoch=50, epochs=10, validati
on_data=test, verbose=1)
```

```
Epoch 1/10
50/50 [==============================] - 138s 3s/step - loss: 1.2823 - accuracy: 0.4
833 - val_loss: 0.9981 - val_accuracy: 0.6157
Epoch 2/10
50/50 [==============================] - 140s 3s/step - loss: 0.9582 - accuracy: 0.6
215 - val_loss: 0.9331 - val_accuracy: 0.6407
Epoch 3/10
50/50 [==============================] - 133s 3s/step - loss: 0.8433 - accuracy: 0.6
860 - val_loss: 0.7722 - val_accuracy: 0.7263
Epoch 4/10
50/50 [==============================] - 140s 3s/step - loss: 0.7264 - accuracy: 0.7
300 - val_loss: 0.6657 - val_accuracy: 0.7597
Epoch 5/10
50/50 [==============================] - 143s 3s/step - loss: 0.6635 - accuracy: 0.7
592 - val_loss: 0.6009 - val_accuracy: 0.7873
Epoch 6/10
50/50 [==============================] - 132s 3s/step - loss: 0.5851 - accuracy: 0.7
871 - val_loss: 0.5387 - val_accuracy: 0.8157
Epoch 7/10
50/50 [==============================] - 135s 3s/step - loss: 0.5527 - accuracy: 0.7
988 - val_loss: 0.5043 - val_accuracy: 0.8250
Epoch 8/10
50/50 [==============================] - 138s 3s/step - loss: 0.5154 - accuracy: 0.8
154 - val_loss: 0.5409 - val_accuracy: 0.8023
Epoch 9/10
50/50 [==============================] - 135s 3s/step - loss: 0.4963 - accuracy: 0.8
194 - val_loss: 0.5260 - val_accuracy: 0.8220
Epoch 10/10
50/50 [==============================] - 131s 3s/step - loss: 0.4664 - accuracy: 0.8
368 - val_loss: 0.5435 - val_accuracy: 0.8117
```

In [ ]:

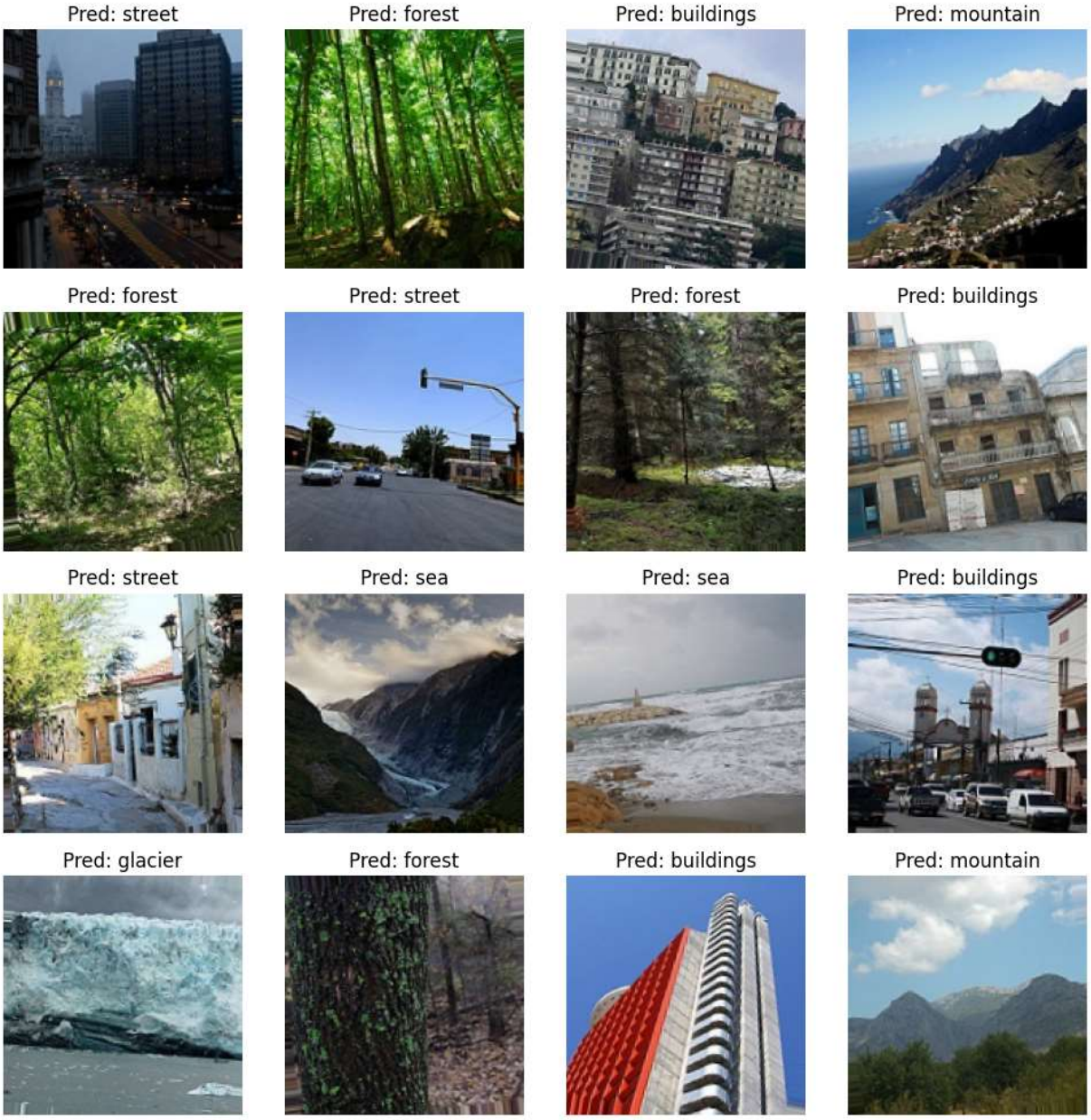
```
plt.plot(cnn.history['accuracy'])
plt.plot(cnn.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='lower right')
plt.show()

plt.plot(cnn.history['loss'])
plt.plot(cnn.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='lower right')
plt.show()

cwd = os.getcwd() + "\\archive\\seg_pred\\seg_pred\\"
img_names = os.listdir(cwd)
random_img_names = random.sample(img_names, 5)

class_dict = {0: 'buildings', 1: 'forest', 2: 'glacier', 3: 'mountain', 4: 'sea', 5
```


Model accuracy
Accuracy
Epoch
Train
Test
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0
2
4
6
8


Model loss
Loss
Epoch
Train
Test
1.3
1.2
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0
2
4
6
8

In [ ]:
```
pred_url = os.getcwd() + "\\archive\\seg_pred"


datagen = ImageDataGenerator(
    rescale=1./255,  # rescale pixel values to be between 0 and 1
    rotation_range=10,  # randomly rotate images by up to 20 degrees
    horizontal_flip=True)  # randomly flip images horizontally

pred = datagen.flow_from_directory(pred_url, target_size=(150, 150), batch_size=256


import matplotlib.pyplot as plt
import numpy as np

# get the class labels
class_labels = list(train.class_indices.keys())

# get a batch of images and their predictions
x, _ = next(pred)
y_pred = model.predict(x)

# display the images and their predictions
fig, axs = plt.subplots(4, 4, figsize=(10, 10))
for i, ax in enumerate(axs.flatten()):
    # show the image
    ax.imshow(x[i])
    ax.axis('off')

    # show the predicted label
    pred_label = class_labels[np.argmax(y_pred[i])]
    ax.set_title(f'Pred: {pred_label}')

plt.tight_layout()
plt.show()
```

```
Found 7301 images belonging to 1 classes.
8/8 [==============================] - 1s 75ms/step
```



Now to use transfer learning.

In [ ]:
```
import tensorflow as tf

base_model = tf.keras.applications.ResNet50(weights='imagenet', include_top = False

x = base_model.output
x = tf.keras.layers.GlobalAveragePooling2D()(x)
x = tf.keras.layers.Dense(1024, activation='relu')(x)
x = tf.keras.layers.Dropout(0.5)(x)
x = tf.keras.layers.Dense(1024, activation='relu')(x)
x = tf.keras.layers.Dropout(0.5)(x)
x = tf.keras.layers.Dense(1024, activation='relu')(x)
x = tf.keras.layers.Dropout(0.5)(x)
x = tf.keras.layers.Dense(512, activation='relu')(x)
x = tf.keras.layers.Dropout(0.5)(x)
preds = tf.keras.layers.Dense(6, activation ='softmax')(x)
```

```
model = tf.keras.models.Model(inputs=base_model.input, outputs=preds)
print(model.summary())
```

```
Model: "model_2"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_4 (InputLayer)           [(None, None, None,  0           []
                                 3)]

 conv1_pad (ZeroPadding2D)      (None, None, None,   0           ['input_4[0][0]']
                                3)

 conv1_conv (Conv2D)            (None, None, None,   9472        ['conv1_pad[0][0]']
                                64)

 conv1_bn (BatchNormalization)  (None, None, None,   256         ['conv1_conv[0]
[0]']
                                64)

 conv1_relu (Activation)        (None, None, None,   0           ['conv1_bn[0][0]']
                                64)

 pool1_pad (ZeroPadding2D)      (None, None, None,   0           ['conv1_relu[0]
[0]']
                                64)

 pool1_pool (MaxPooling2D)      (None, None, None,   0           ['pool1_pad[0][0]']
                                64)

 conv2_block1_1_conv (Conv2D)   (None, None, None,   4160        ['pool1_pool[0]
[0]']
                                64)

 conv2_block1_1_bn (BatchNormal  (None, None, None,   256        ['conv2_block1_1_co
nv[0][0]']
 ization)                       64)

 conv2_block1_1_relu (Activatio  (None, None, None,   0          ['conv2_block1_1_bn
[0][0]']
 n)                             64)

 conv2_block1_2_conv (Conv2D)   (None, None, None,   36928       ['conv2_block1_1_re
lu[0][0]']
                                64)

 conv2_block1_2_bn (BatchNormal  (None, None, None,   256        ['conv2_block1_2_co
nv[0][0]']
 ization)                       64)

 conv2_block1_2_relu (Activatio  (None, None, None,   0          ['conv2_block1_2_bn
[0][0]']
 n)                             64)

 conv2_block1_0_conv (Conv2D)   (None, None, None,   16640       ['pool1_pool[0]
[0]']
                                256)
```

```
 conv2_block1_3_conv (Conv2D)   (None, None, None,   16640       ['conv2_block1_2_re
lu[0][0]']
                                256)

 conv2_block1_0_bn (BatchNormal  (None, None, None,   1024       ['conv2_block1_0_co
nv[0][0]']
 ization)                       256)

 conv2_block1_3_bn (BatchNormal  (None, None, None,   1024       ['conv2_block1_3_co
nv[0][0]']
 ization)                       256)

 conv2_block1_add (Add)         (None, None, None,   0           ['conv2_block1_0_bn
[0][0]',
                                256)                             'conv2_block1_3_bn
[0][0]']

 conv2_block1_out (Activation)  (None, None, None,   0           ['conv2_block1_add
[0][0]']
                                256)

 conv2_block2_1_conv (Conv2D)   (None, None, None,   16448       ['conv2_block1_out
[0][0]']
                                64)

 conv2_block2_1_bn (BatchNormal  (None, None, None,   256        ['conv2_block2_1_co
nv[0][0]']
 ization)                       64)

 conv2_block2_1_relu (Activatio  (None, None, None,   0          ['conv2_block2_1_bn
[0][0]']
 n)                             64)

 conv2_block2_2_conv (Conv2D)   (None, None, None,   36928       ['conv2_block2_1_re
lu[0][0]']
                                64)

 conv2_block2_2_bn (BatchNormal  (None, None, None,   256        ['conv2_block2_2_co
nv[0][0]']
 ization)                       64)

 conv2_block2_2_relu (Activatio  (None, None, None,   0          ['conv2_block2_2_bn
[0][0]']
 n)                             64)

 conv2_block2_3_conv (Conv2D)   (None, None, None,   16640       ['conv2_block2_2_re
lu[0][0]']
                                256)

 conv2_block2_3_bn (BatchNormal  (None, None, None,   1024       ['conv2_block2_3_co
nv[0][0]']
 ization)                       256)

 conv2_block2_add (Add)         (None, None, None,   0           ['conv2_block1_out
[0][0]',
```

```
                                256)                                'conv2_block2_3_bn
[0][0]']

 conv2_block2_out (Activation)  (None, None, None,   0           ['conv2_block2_add
[0][0]']
                                256)

 conv2_block3_1_conv (Conv2D)   (None, None, None,   16448       ['conv2_block2_out
[0][0]']
                                64)

 conv2_block3_1_bn (BatchNormal  (None, None, None,   256        ['conv2_block3_1_co
nv[0][0]']
 ization)                       64)

 conv2_block3_1_relu (Activatio  (None, None, None,   0          ['conv2_block3_1_bn
[0][0]']
 n)                             64)

 conv2_block3_2_conv (Conv2D)   (None, None, None,   36928       ['conv2_block3_1_re
lu[0][0]']
                                64)

 conv2_block3_2_bn (BatchNormal  (None, None, None,   256        ['conv2_block3_2_co
nv[0][0]']
 ization)                       64)

 conv2_block3_2_relu (Activatio  (None, None, None,   0          ['conv2_block3_2_bn
[0][0]']
 n)                             64)

 conv2_block3_3_conv (Conv2D)   (None, None, None,   16640       ['conv2_block3_2_re
lu[0][0]']
                                256)

 conv2_block3_3_bn (BatchNormal  (None, None, None,   1024       ['conv2_block3_3_co
nv[0][0]']
 ization)                       256)

 conv2_block3_add (Add)         (None, None, None,   0           ['conv2_block2_out
[0][0]',
                                256)                                'conv2_block3_3_bn
[0][0]']

 conv2_block3_out (Activation)  (None, None, None,   0           ['conv2_block3_add
[0][0]']
                                256)

 conv3_block1_1_conv (Conv2D)   (None, None, None,   32896       ['conv2_block3_out
[0][0]']
                                128)

 conv3_block1_1_bn (BatchNormal  (None, None, None,   512        ['conv3_block1_1_co
nv[0][0]']
 ization)                       128)
```

```
 conv3_block1_1_relu (Activatio  (None, None, None,   0          ['conv3_block1_1_bn
[0][0]']
 n)                             128)

 conv3_block1_2_conv (Conv2D)   (None, None, None,   147584      ['conv3_block1_1_re
lu[0][0]']
                                128)

 conv3_block1_2_bn (BatchNormal  (None, None, None,   512        ['conv3_block1_2_co
nv[0][0]']
 ization)                       128)

 conv3_block1_2_relu (Activatio  (None, None, None,   0          ['conv3_block1_2_bn
[0][0]']
 n)                             128)

 conv3_block1_0_conv (Conv2D)   (None, None, None,   131584      ['conv2_block3_out
[0][0]']
                                512)

 conv3_block1_3_conv (Conv2D)   (None, None, None,   66048       ['conv3_block1_2_re
lu[0][0]']
                                512)

 conv3_block1_0_bn (BatchNormal  (None, None, None,   2048       ['conv3_block1_0_co
nv[0][0]']
 ization)                       512)

 conv3_block1_3_bn (BatchNormal  (None, None, None,   2048       ['conv3_block1_3_co
nv[0][0]']
 ization)                       512)

 conv3_block1_add (Add)         (None, None, None,   0          ['conv3_block1_0_bn
[0][0]',
                                512)                             'conv3_block1_3_bn
[0][0]']

 conv3_block1_out (Activation)  (None, None, None,   0          ['conv3_block1_add
[0][0]']
                                512)

 conv3_block2_1_conv (Conv2D)   (None, None, None,   65664       ['conv3_block1_out
[0][0]']
                                128)

 conv3_block2_1_bn (BatchNormal  (None, None, None,   512        ['conv3_block2_1_co
nv[0][0]']
 ization)                       128)

 conv3_block2_1_relu (Activatio  (None, None, None,   0          ['conv3_block2_1_bn
[0][0]']
 n)                             128)

 conv3_block2_2_conv (Conv2D)   (None, None, None,   147584      ['conv3_block2_1_re
lu[0][0]']
                                128)
```

```
 conv3_block2_2_bn (BatchNormal  (None, None, None,   512        ['conv3_block2_2_co
nv[0][0]']
 ization)                       128)

 conv3_block2_2_relu (Activatio  (None, None, None,   0          ['conv3_block2_2_bn
[0][0]']
 n)                             128)

 conv3_block2_3_conv (Conv2D)   (None, None, None,   66048       ['conv3_block2_2_re
lu[0][0]']
                                512)

 conv3_block2_3_bn (BatchNormal  (None, None, None,   2048       ['conv3_block2_3_co
nv[0][0]']
 ization)                       512)

 conv3_block2_add (Add)         (None, None, None,   0           ['conv3_block1_out
[0][0]',
                                512)                             'conv3_block2_3_bn
[0][0]']

 conv3_block2_out (Activation)  (None, None, None,   0           ['conv3_block2_add
[0][0]']
                                512)

 conv3_block3_1_conv (Conv2D)   (None, None, None,   65664       ['conv3_block2_out
[0][0]']
                                128)

 conv3_block3_1_bn (BatchNormal  (None, None, None,   512        ['conv3_block3_1_co
nv[0][0]']
 ization)                       128)

 conv3_block3_1_relu (Activatio  (None, None, None,   0          ['conv3_block3_1_bn
[0][0]']
 n)                             128)

 conv3_block3_2_conv (Conv2D)   (None, None, None,   147584      ['conv3_block3_1_re
lu[0][0]']
                                128)

 conv3_block3_2_bn (BatchNormal  (None, None, None,   512        ['conv3_block3_2_co
nv[0][0]']
 ization)                       128)

 conv3_block3_2_relu (Activatio  (None, None, None,   0          ['conv3_block3_2_bn
[0][0]']
 n)                             128)

 conv3_block3_3_conv (Conv2D)   (None, None, None,   66048       ['conv3_block3_2_re
lu[0][0]']
                                512)

 conv3_block3_3_bn (BatchNormal  (None, None, None,   2048       ['conv3_block3_3_co
nv[0][0]']
```

```
 ization)                       512)

 conv3_block3_add (Add)         (None, None, None,   0           ['conv3_block2_out
[0][0]',
                                512)                              'conv3_block3_3_bn
[0][0]']

 conv3_block3_out (Activation)  (None, None, None,   0           ['conv3_block3_add
[0][0]']
                                512)

 conv3_block4_1_conv (Conv2D)   (None, None, None,   65664       ['conv3_block3_out
[0][0]']
                                128)

 conv3_block4_1_bn (BatchNormal  (None, None, None,   512        ['conv3_block4_1_co
nv[0][0]']
 ization)                       128)

 conv3_block4_1_relu (Activatio  (None, None, None,   0          ['conv3_block4_1_bn
[0][0]']
 n)                             128)

 conv3_block4_2_conv (Conv2D)   (None, None, None,   147584      ['conv3_block4_1_re
lu[0][0]']
                                128)

 conv3_block4_2_bn (BatchNormal  (None, None, None,   512        ['conv3_block4_2_co
nv[0][0]']
 ization)                       128)

 conv3_block4_2_relu (Activatio  (None, None, None,   0          ['conv3_block4_2_bn
[0][0]']
 n)                             128)

 conv3_block4_3_conv (Conv2D)   (None, None, None,   66048       ['conv3_block4_2_re
lu[0][0]']
                                512)

 conv3_block4_3_bn (BatchNormal  (None, None, None,   2048       ['conv3_block4_3_co
nv[0][0]']
 ization)                       512)

 conv3_block4_add (Add)         (None, None, None,   0           ['conv3_block3_out
[0][0]',
                                512)                              'conv3_block4_3_bn
[0][0]']

 conv3_block4_out (Activation)  (None, None, None,   0           ['conv3_block4_add
[0][0]']
                                512)

 conv4_block1_1_conv (Conv2D)   (None, None, None,   131328      ['conv3_block4_out
[0][0]']
                                256)
```

```
 conv4_block1_1_bn (BatchNormal  (None, None, None,   1024       ['conv4_block1_1_co
nv[0][0]']
 ization)                       256)

 conv4_block1_1_relu (Activatio  (None, None, None,   0          ['conv4_block1_1_bn
[0][0]']
 n)                             256)

 conv4_block1_2_conv (Conv2D)   (None, None, None,   590080      ['conv4_block1_1_re
lu[0][0]']
                                256)

 conv4_block1_2_bn (BatchNormal  (None, None, None,   1024       ['conv4_block1_2_co
nv[0][0]']
 ization)                       256)

 conv4_block1_2_relu (Activatio  (None, None, None,   0          ['conv4_block1_2_bn
[0][0]']
 n)                             256)

 conv4_block1_0_conv (Conv2D)   (None, None, None,   525312      ['conv3_block4_out
[0][0]']
                                1024)

 conv4_block1_3_conv (Conv2D)   (None, None, None,   263168      ['conv4_block1_2_re
lu[0][0]']
                                1024)

 conv4_block1_0_bn (BatchNormal  (None, None, None,   4096       ['conv4_block1_0_co
nv[0][0]']
 ization)                       1024)

 conv4_block1_3_bn (BatchNormal  (None, None, None,   4096       ['conv4_block1_3_co
nv[0][0]']
 ization)                       1024)

 conv4_block1_add (Add)         (None, None, None,   0          ['conv4_block1_0_bn
[0][0]',
                                1024)                            'conv4_block1_3_bn
[0][0]']

 conv4_block1_out (Activation)  (None, None, None,   0          ['conv4_block1_add
[0][0]']
                                1024)

 conv4_block2_1_conv (Conv2D)   (None, None, None,   262400      ['conv4_block1_out
[0][0]']
                                256)

 conv4_block2_1_bn (BatchNormal  (None, None, None,   1024       ['conv4_block2_1_co
nv[0][0]']
 ization)                       256)

 conv4_block2_1_relu (Activatio  (None, None, None,   0          ['conv4_block2_1_bn
[0][0]']
 n)                             256)
```

```
 conv4_block2_2_conv (Conv2D)   (None, None, None,   590080      ['conv4_block2_1_re
lu[0][0]']
                                256)

 conv4_block2_2_bn (BatchNormal  (None, None, None,   1024       ['conv4_block2_2_co
nv[0][0]']
 ization)                       256)

 conv4_block2_2_relu (Activatio  (None, None, None,   0          ['conv4_block2_2_bn
[0][0]']
 n)                             256)

 conv4_block2_3_conv (Conv2D)   (None, None, None,   263168      ['conv4_block2_2_re
lu[0][0]']
                                1024)

 conv4_block2_3_bn (BatchNormal  (None, None, None,   4096       ['conv4_block2_3_co
nv[0][0]']
 ization)                       1024)

 conv4_block2_add (Add)         (None, None, None,   0          ['conv4_block1_out
[0][0]',
                                1024)                            'conv4_block2_3_bn
[0][0]']

 conv4_block2_out (Activation)  (None, None, None,   0          ['conv4_block2_add
[0][0]']
                                1024)

 conv4_block3_1_conv (Conv2D)   (None, None, None,   262400      ['conv4_block2_out
[0][0]']
                                256)

 conv4_block3_1_bn (BatchNormal  (None, None, None,   1024       ['conv4_block3_1_co
nv[0][0]']
 ization)                       256)

 conv4_block3_1_relu (Activatio  (None, None, None,   0          ['conv4_block3_1_bn
[0][0]']
 n)                             256)

 conv4_block3_2_conv (Conv2D)   (None, None, None,   590080      ['conv4_block3_1_re
lu[0][0]']
                                256)

 conv4_block3_2_bn (BatchNormal  (None, None, None,   1024       ['conv4_block3_2_co
nv[0][0]']
 ization)                       256)

 conv4_block3_2_relu (Activatio  (None, None, None,   0          ['conv4_block3_2_bn
[0][0]']
 n)                             256)

 conv4_block3_3_conv (Conv2D)   (None, None, None,   263168      ['conv4_block3_2_re
lu[0][0]']
```

```
                                1024)

 conv4_block3_3_bn (BatchNormal  (None, None, None,   4096       ['conv4_block3_3_co
nv[0][0]']
 ization)                       1024)

 conv4_block3_add (Add)         (None, None, None,   0          ['conv4_block2_out
[0][0]',
                                1024)                            'conv4_block3_3_bn
[0][0]']

 conv4_block3_out (Activation)  (None, None, None,   0          ['conv4_block3_add
[0][0]']
                                1024)

 conv4_block4_1_conv (Conv2D)   (None, None, None,   262400     ['conv4_block3_out
[0][0]']
                                256)

 conv4_block4_1_bn (BatchNormal  (None, None, None,   1024       ['conv4_block4_1_co
nv[0][0]']
 ization)                       256)

 conv4_block4_1_relu (Activatio  (None, None, None,   0          ['conv4_block4_1_bn
[0][0]']
 n)                             256)

 conv4_block4_2_conv (Conv2D)   (None, None, None,   590080     ['conv4_block4_1_re
lu[0][0]']
                                256)

 conv4_block4_2_bn (BatchNormal  (None, None, None,   1024       ['conv4_block4_2_co
nv[0][0]']
 ization)                       256)

 conv4_block4_2_relu (Activatio  (None, None, None,   0          ['conv4_block4_2_bn
[0][0]']
 n)                             256)

 conv4_block4_3_conv (Conv2D)   (None, None, None,   263168     ['conv4_block4_2_re
lu[0][0]']
                                1024)

 conv4_block4_3_bn (BatchNormal  (None, None, None,   4096       ['conv4_block4_3_co
nv[0][0]']
 ization)                       1024)

 conv4_block4_add (Add)         (None, None, None,   0          ['conv4_block3_out
[0][0]',
                                1024)                            'conv4_block4_3_bn
[0][0]']

 conv4_block4_out (Activation)  (None, None, None,   0          ['conv4_block4_add
[0][0]']
                                1024)
```

```
 conv4_block5_1_conv (Conv2D)   (None, None, None,   262400      ['conv4_block4_out
[0][0]']
                                256)

 conv4_block5_1_bn (BatchNormal  (None, None, None,   1024       ['conv4_block5_1_co
nv[0][0]']
 ization)                       256)

 conv4_block5_1_relu (Activatio  (None, None, None,   0          ['conv4_block5_1_bn
[0][0]']
 n)                             256)

 conv4_block5_2_conv (Conv2D)   (None, None, None,   590080      ['conv4_block5_1_re
lu[0][0]']
                                256)

 conv4_block5_2_bn (BatchNormal  (None, None, None,   1024       ['conv4_block5_2_co
nv[0][0]']
 ization)                       256)

 conv4_block5_2_relu (Activatio  (None, None, None,   0          ['conv4_block5_2_bn
[0][0]']
 n)                             256)

 conv4_block5_3_conv (Conv2D)   (None, None, None,   263168      ['conv4_block5_2_re
lu[0][0]']
                                1024)

 conv4_block5_3_bn (BatchNormal  (None, None, None,   4096       ['conv4_block5_3_co
nv[0][0]']
 ization)                       1024)

 conv4_block5_add (Add)         (None, None, None,   0          ['conv4_block4_out
[0][0]',
                                1024)                            'conv4_block5_3_bn
[0][0]']

 conv4_block5_out (Activation)  (None, None, None,   0          ['conv4_block5_add
[0][0]']
                                1024)

 conv4_block6_1_conv (Conv2D)   (None, None, None,   262400      ['conv4_block5_out
[0][0]']
                                256)

 conv4_block6_1_bn (BatchNormal  (None, None, None,   1024       ['conv4_block6_1_co
nv[0][0]']
 ization)                       256)

 conv4_block6_1_relu (Activatio  (None, None, None,   0          ['conv4_block6_1_bn
[0][0]']
 n)                             256)

 conv4_block6_2_conv (Conv2D)   (None, None, None,   590080      ['conv4_block6_1_re
lu[0][0]']
                                256)
```

```
 conv4_block6_2_bn (BatchNormal  (None, None, None,   1024       ['conv4_block6_2_co
nv[0][0]']
 ization)                       256)

 conv4_block6_2_relu (Activatio  (None, None, None,   0          ['conv4_block6_2_bn
[0][0]']
 n)                             256)

 conv4_block6_3_conv (Conv2D)   (None, None, None,   263168      ['conv4_block6_2_re
lu[0][0]']
                                1024)

 conv4_block6_3_bn (BatchNormal  (None, None, None,   4096       ['conv4_block6_3_co
nv[0][0]']
 ization)                       1024)

 conv4_block6_add (Add)         (None, None, None,   0           ['conv4_block5_out
[0][0]',
                                1024)                            'conv4_block6_3_bn
[0][0]']

 conv4_block6_out (Activation)  (None, None, None,   0           ['conv4_block6_add
[0][0]']
                                1024)

 conv5_block1_1_conv (Conv2D)   (None, None, None,   524800      ['conv4_block6_out
[0][0]']
                                512)

 conv5_block1_1_bn (BatchNormal  (None, None, None,   2048       ['conv5_block1_1_co
nv[0][0]']
 ization)                       512)

 conv5_block1_1_relu (Activatio  (None, None, None,   0          ['conv5_block1_1_bn
[0][0]']
 n)                             512)

 conv5_block1_2_conv (Conv2D)   (None, None, None,   2359808     ['conv5_block1_1_re
lu[0][0]']
                                512)

 conv5_block1_2_bn (BatchNormal  (None, None, None,   2048       ['conv5_block1_2_co
nv[0][0]']
 ization)                       512)

 conv5_block1_2_relu (Activatio  (None, None, None,   0          ['conv5_block1_2_bn
[0][0]']
 n)                             512)

 conv5_block1_0_conv (Conv2D)   (None, None, None,   2099200     ['conv4_block6_out
[0][0]']
                                2048)

 conv5_block1_3_conv (Conv2D)   (None, None, None,   1050624     ['conv5_block1_2_re
lu[0][0]']
```

```
                                2048)

 conv5_block1_0_bn (BatchNormal  (None, None, None,   8192       ['conv5_block1_0_co
nv[0][0]']
 ization)                       2048)

 conv5_block1_3_bn (BatchNormal  (None, None, None,   8192       ['conv5_block1_3_co
nv[0][0]']
 ization)                       2048)

 conv5_block1_add (Add)         (None, None, None,   0           ['conv5_block1_0_bn
[0][0]',
                                2048)                             'conv5_block1_3_bn
[0][0]']

 conv5_block1_out (Activation)  (None, None, None,   0           ['conv5_block1_add
[0][0]']
                                2048)

 conv5_block2_1_conv (Conv2D)   (None, None, None,   1049088     ['conv5_block1_out
[0][0]']
                                512)

 conv5_block2_1_bn (BatchNormal  (None, None, None,   2048       ['conv5_block2_1_co
nv[0][0]']
 ization)                       512)

 conv5_block2_1_relu (Activatio  (None, None, None,   0          ['conv5_block2_1_bn
[0][0]']
 n)                             512)

 conv5_block2_2_conv (Conv2D)   (None, None, None,   2359808     ['conv5_block2_1_re
lu[0][0]']
                                512)

 conv5_block2_2_bn (BatchNormal  (None, None, None,   2048       ['conv5_block2_2_co
nv[0][0]']
 ization)                       512)

 conv5_block2_2_relu (Activatio  (None, None, None,   0          ['conv5_block2_2_bn
[0][0]']
 n)                             512)

 conv5_block2_3_conv (Conv2D)   (None, None, None,   1050624     ['conv5_block2_2_re
lu[0][0]']
                                2048)

 conv5_block2_3_bn (BatchNormal  (None, None, None,   8192       ['conv5_block2_3_co
nv[0][0]']
 ization)                       2048)

 conv5_block2_add (Add)         (None, None, None,   0           ['conv5_block1_out
[0][0]',
                                2048)                             'conv5_block2_3_bn
[0][0]']
```

```
 conv5_block2_out (Activation)  (None, None, None,   0           ['conv5_block2_add
[0][0]']
                                2048)

 conv5_block3_1_conv (Conv2D)   (None, None, None,   1049088     ['conv5_block2_out
[0][0]']
                                512)

 conv5_block3_1_bn (BatchNormal  (None, None, None,   2048       ['conv5_block3_1_co
nv[0][0]']
 ization)                       512)

 conv5_block3_1_relu (Activatio  (None, None, None,   0          ['conv5_block3_1_bn
[0][0]']
 n)                             512)

 conv5_block3_2_conv (Conv2D)   (None, None, None,   2359808     ['conv5_block3_1_re
lu[0][0]']
                                512)

 conv5_block3_2_bn (BatchNormal  (None, None, None,   2048       ['conv5_block3_2_co
nv[0][0]']
 ization)                       512)

 conv5_block3_2_relu (Activatio  (None, None, None,   0          ['conv5_block3_2_bn
[0][0]']
 n)                             512)

 conv5_block3_3_conv (Conv2D)   (None, None, None,   1050624     ['conv5_block3_2_re
lu[0][0]']
                                2048)

 conv5_block3_3_bn (BatchNormal  (None, None, None,   8192       ['conv5_block3_3_co
nv[0][0]']
 ization)                       2048)

 conv5_block3_add (Add)         (None, None, None,   0           ['conv5_block2_out
[0][0]',
                                2048)                            'conv5_block3_3_bn
[0][0]']

 conv5_block3_out (Activation)  (None, None, None,   0           ['conv5_block3_add
[0][0]']
                                2048)

 global_average_pooling2d_3 (Gl  (None, 2048)        0           ['conv5_block3_out
[0][0]']
 obalAveragePooling2D)

 dense_35 (Dense)               (None, 1024)         2098176     ['global_average_po
oling2d_3[0][0
                                                                 ]']

 dropout_14 (Dropout)           (None, 1024)         0           ['dense_35[0][0]']

 dense_36 (Dense)               (None, 1024)         1049600     ['dropout_14[0]
```

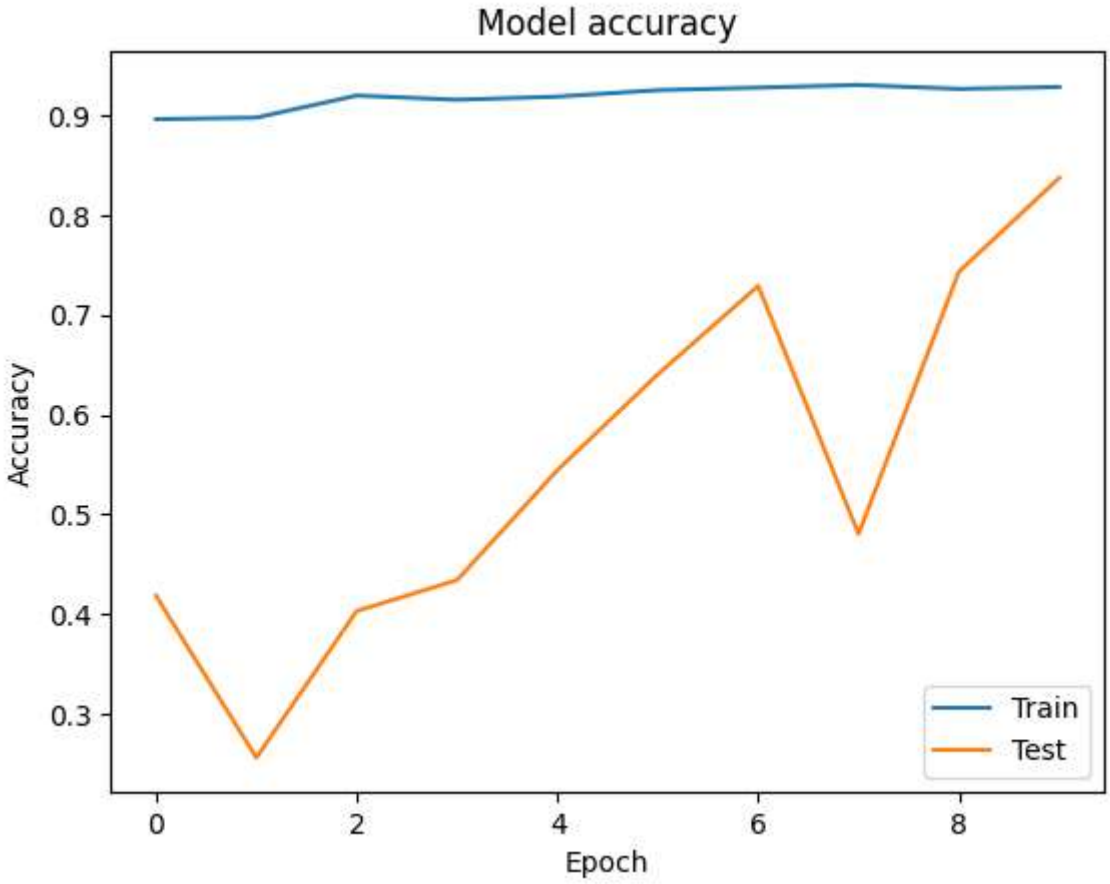
```
[0]']

 dropout_15 (Dropout)           (None, 1024)         0           ['dense_36[0][0]']

 dense_37 (Dense)               (None, 1024)         1049600     ['dropout_15[0]
[0]']

 dropout_16 (Dropout)           (None, 1024)         0           ['dense_37[0][0]']

 dense_38 (Dense)               (None, 512)          524800      ['dropout_16[0]
[0]']

 dropout_17 (Dropout)           (None, 512)          0           ['dense_38[0][0]']

 dense_39 (Dense)               (None, 6)            3078        ['dropout_17[0]
[0]']

==================================================================================
==============
Total params: 28,312,966
Trainable params: 28,259,846
Non-trainable params: 53,120
__________________________________________________________________________________
______________
None
```

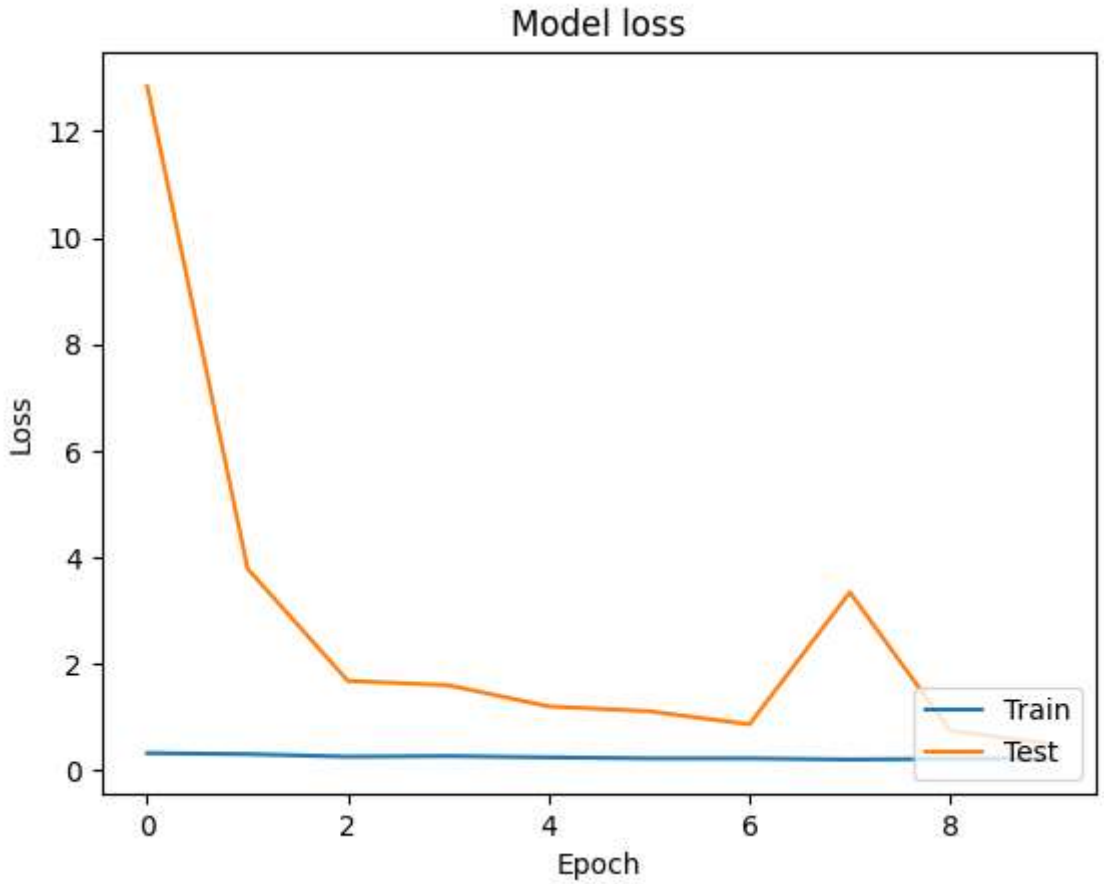
In [ ]:
```
datagen = ImageDataGenerator(
    rescale=1./255,  # rescale pixel values to be between 0 and 1
    rotation_range=10,  # randomly rotate images by up to 20 degrees
    horizontal_flip=True)  # randomly flip images horizontally

train = datagen.flow_from_directory(train_url, target_size=(150, 150), batch_size=1
test = datagen.flow_from_directory(test_url, target_size=(150, 150), batch_size=128

callback = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=3)

if os.path.exists('models/image_classification_model_5.h5'):
    model = tf.keras.models.load_model('models/image_classification_model_4.h5')

else:
    model.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['accu
    cnn = model.fit(train, steps_per_epoch=50, epochs = 10, validation_data=test, v
    model.save('models/image_classification_model_4.h5')
```

```
Found 14034 images belonging to 6 classes.
Found 3000 images belonging to 6 classes.
Epoch 1/10
50/50 [==============================] - 661s 13s/step - loss: 0.3220 - accuracy: 0.
8961 - val_loss: 12.8451 - val_accuracy: 0.4183
Epoch 2/10
50/50 [==============================] - 694s 14s/step - loss: 0.3035 - accuracy: 0.
8977 - val_loss: 3.7855 - val_accuracy: 0.2560
Epoch 3/10
50/50 [==============================] - 632s 13s/step - loss: 0.2555 - accuracy: 0.
9198 - val_loss: 1.6775 - val_accuracy: 0.4030
Epoch 4/10
50/50 [==============================] - 653s 13s/step - loss: 0.2665 - accuracy: 0.
9156 - val_loss: 1.5976 - val_accuracy: 0.4340
Epoch 5/10
50/50 [==============================] - 690s 14s/step - loss: 0.2424 - accuracy: 0.
9187 - val_loss: 1.1986 - val_accuracy: 0.5443
Epoch 6/10
50/50 [==============================] - 599s 12s/step - loss: 0.2225 - accuracy: 0.
9252 - val_loss: 1.1079 - val_accuracy: 0.6403
Epoch 7/10
50/50 [==============================] - 568s 11s/step - loss: 0.2231 - accuracy: 0.
9278 - val_loss: 0.8623 - val_accuracy: 0.7290
Epoch 8/10
50/50 [==============================] - 597s 12s/step - loss: 0.2018 - accuracy: 0.
9305 - val_loss: 3.3420 - val_accuracy: 0.4803
Epoch 9/10
50/50 [==============================] - 573s 11s/step - loss: 0.2140 - accuracy: 0.
9265 - val_loss: 0.7422 - val_accuracy: 0.7433
Epoch 10/10
50/50 [==============================] - 581s 12s/step - loss: 0.2206 - accuracy: 0.
9284 - val_loss: 0.4969 - val_accuracy: 0.8373
```

In [ ]:

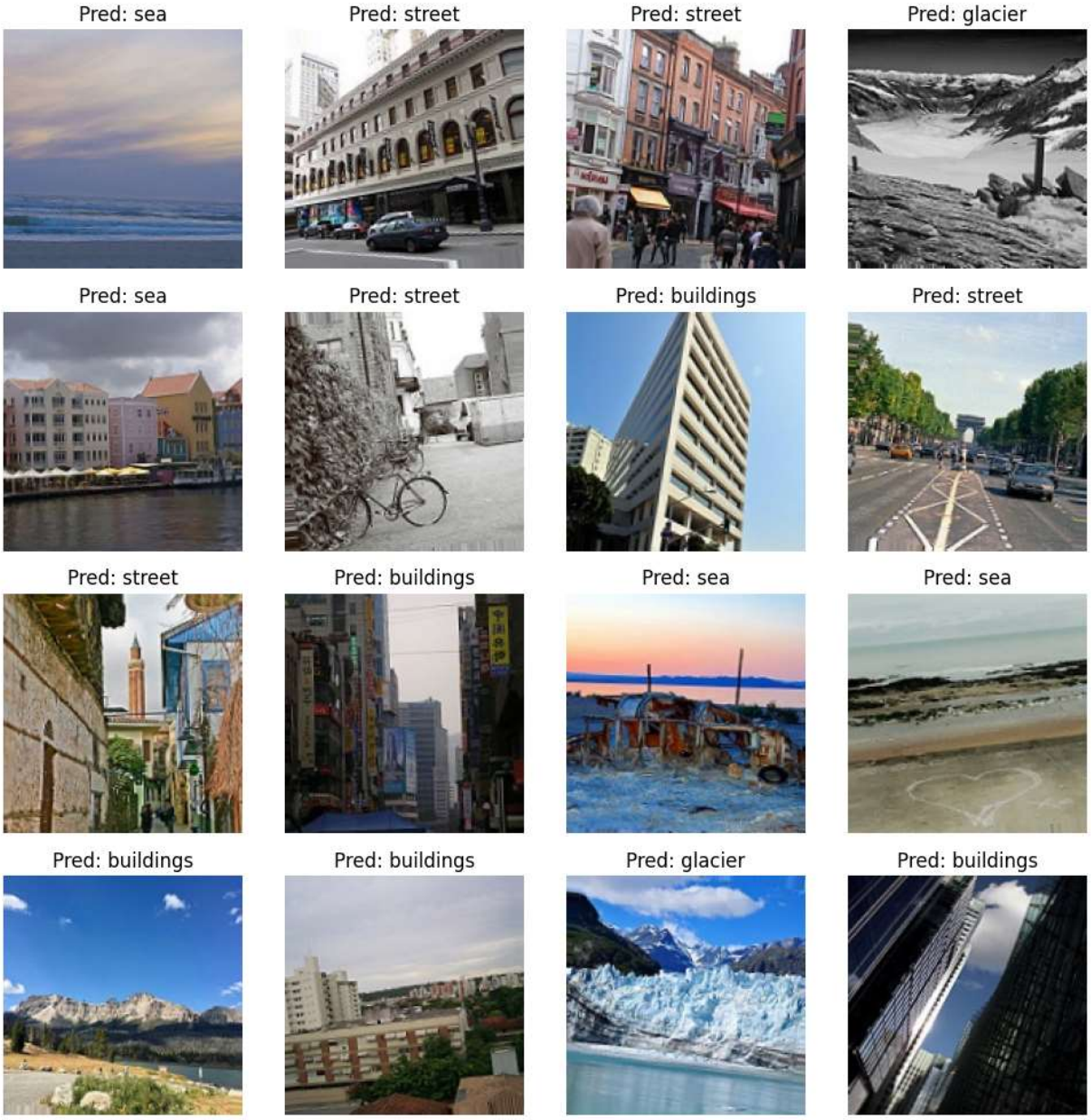
```
plt.plot(cnn.history['accuracy'])
plt.plot(cnn.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='lower right')
plt.show()

plt.plot(cnn.history['loss'])
plt.plot(cnn.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='lower right')
plt.show()
```


Model accuracy
Accuracy
Epoch
Train
Test
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
2
4
6
8


Model loss
Loss
Epoch
Train
Test
12
10
8
6
4
2
0
0
2
4
6
8

In [ ]:
```
pred_url = os.getcwd() + "\\archive\\seg_pred"


datagen = ImageDataGenerator(
    rescale=1./255,  # rescale pixel values to be between 0 and 1
    rotation_range=10,  # randomly rotate images by up to 20 degrees
    horizontal_flip=True)  # randomly flip images horizontally

pred = datagen.flow_from_directory(pred_url, target_size=(150, 150), batch_size=256


import matplotlib.pyplot as plt
import numpy as np

# get the class labels
class_labels = list(train.class_indices.keys())

# get a batch of images and their predictions
x, _ = next(pred)
y_pred = model.predict(x)

# display the images and their predictions
fig, axs = plt.subplots(4, 4, figsize=(10, 10))
for i, ax in enumerate(axs.flatten()):
    # show the image
    ax.imshow(x[i])
    ax.axis('off')

    # show the predicted label
    pred_label = class_labels[np.argmax(y_pred[i])]
    ax.set_title(f'Pred: {pred_label}')

plt.tight_layout()
plt.show()
```

```
Found 7301 images belonging to 1 classes.
8/8 [==============================] - 5s 559ms/step
```



Model Evaluation

As can be seen in all the models above, the accuracy for the sequential model with color images and 4 conv2d layers has the lowest accuracy.
With just a simple change in using grayscale images, the accuracy jumps 10% with it going from 50 to 62.

The deep learning model seems to work better with less layers since the accuracy for the 3 conv2d layer model was around 65%, far higher than the 4conv2d layer model. Note that there were no other changes made to origrinal architecture, even the color input was kept the same.

The amount of learning parameters as well as the number of images quite affected the training time of these models. It took an hour each to train most of them, especially because some used to stall out for some reason.

The model that performed best was the transfer learning model, with an accuracy of 86%. I could not let it run for longer because of problems with time, since every epoch seemed to take around 10 minutes, and I could not let it run continously external problems. Otherwise I think it would have definitely performed better than what is seen here.