```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
!ls "/content/drive/My Drive"
```

```
'bandicam 2024-10-19 23-59-13-798.zip'  'mahir cs2001030.zip'
'Colab Notebooks'                        monkey
'Data science'                           monkey.ipynb
'dataset sample pic.gdraw'               nlp_no
'Getting started.pdf'                    projectXai.ipynb
 KDDTest-21.txt                         'Untitled drawing (1).gdraw'
 KDDTest+.txt                           'Untitled drawing (2).gdraw'
 KDDTrain+_20Percent.txt                'Untitled drawing.gdraw'
 KDDTrain+.txt
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.image import imread
import cv2
import random
from os import listdir
from sklearn.preprocessing import  LabelBinarizer
from keras.preprocessing import image
from keras.preprocessing.image import img_to_array, array_to_img
from keras.optimizers import Adam
from PIL import Image
from keras.models import Sequential
from keras.layers import BatchNormalization, Conv2D, MaxPooling2D, Activation, Flatten, Dropout, Dense, LeakyReLU
from sklearn.model_selection import train_test_split
```

```
!apt-get install unrar
```

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
unrar is already the newest version (1:6.1.5-1ubuntu0.1).
0 upgraded, 0 newly installed, 0 to remove and 35 not upgraded.
```

```
#/content/drive/MyDrive/Data science/project-13/Data.rar
!unrar x "/content/drive/MyDrive/Data science/project-13/Data.rar" "/content/drive/MyDrive/Data science/project-13/"
```

```
Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24189.jpg  OK
```

```
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24201.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24209.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24214.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24219.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24235.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24236.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24237.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24246.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24266.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24274.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24295.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24301.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24304.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24309.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24311.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/sea/24325.jpg  OK
  Creating    /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street/20066.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street/20067.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street/20069.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street/20070.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street/20075.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street/20079.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street/20080.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street/20084.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street/20088.jpg  OK
  Extracting  /content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/street/20090.jpg  OK
```

```
!ls "/content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset"
```

```
buildings  forest  glacier  mountain  sea  street
```

```
plt.figure(figsize=(11,11))
path = "/content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset/buildings"
for i in range(1,26):
    plt.subplot(5,5,i)
    plt.tight_layout()
    rand_img = imread(path +'/'+ random.choice(sorted(listdir(path))))
    plt.imshow(rand_img)
    plt.title('mountain')
    plt.xlabel(rand_img.shape[1], fontsize = 10)
    plt.ylabel(rand_img.shape[0], fontsize = 10)
```

```
dir = "/content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset"
root_dir = listdir(dir)
image_list, label_list = [], []
```

```
dir
```

```
'/content/drive/MyDrive/Data science/project-13/Data/Intel Image Dataset'
```

```
for directory in root_dir:
  for files in listdir(f"{dir}/{directory}"):
    image_path = f"{dir}/{directory}/{files}"
    image = Image.open(image_path)
    image = image.resize((150,150))
    image = img_to_array(image)
    image_list.append(image)
    label_list.append(directory)
```

```
label_counts = pd.DataFrame(label_list).value_counts()
label_counts
```

```
           count
        0
  glacier    553
 mountain    525
      sea    510
   street    501
   forest    474
buildings    437

dtype: int64
```

```
num_classes = len(label_counts)
num_classes
```

```
6
```

```
np.array(image_list).shape
```

```
(3000, 150, 150, 3)
```

```
label_list = np.array(label_list)
label_list.shape
```

```
(3000,)
```

```
x_train, x_test, y_train, y_test = train_test_split(image_list, label_list, test_size=0.2, random_state = 10)
```

```
x_train = np.array(x_train, dtype=np.float16) / 225.0
x_test = np.array(x_test, dtype=np.float16) / 225.0
x_train = x_train.reshape( -1, 150,150,3)
x_test = x_test.reshape( -1, 150,150,3)
```

```
lb = LabelBinarizer()
y_train = lb.fit_transform(y_train)
y_test = lb.fit_transform(y_test)
print(lb.classes_)
```

```
['buildings' 'forest' 'glacier' 'mountain' 'sea' 'street']
```

```
x_train, x_val, y_train, y_val = train_test_split(x_train, y_train, test_size = 0.2)
```

```
model = Sequential([
        Conv2D(16, kernel_size = (3,3), input_shape = (150,150,3)),
        BatchNormalization(),
        LeakyReLU(),

        Conv2D(32, kernel_size = (3,3)),
        BatchNormalization(),
        LeakyReLU(),
        MaxPooling2D(5,5),

        Conv2D(64, kernel_size = (3,3)),
        BatchNormalization(),
        LeakyReLU(),

        Conv2D(128, kernel_size = (3,3)),
        BatchNormalization(),
        LeakyReLU(),
        MaxPooling2D(5,5),

        Flatten(),

        Dense(64),
        Dropout(rate = 0.2),
        BatchNormalization(),
        LeakyReLU(),

        Dense(32),
        Dropout(rate = 0.2),
        BatchNormalization(),
        LeakyReLU(),

        Dense(16),
        Dropout(rate = 0.2),
        BatchNormalization(),
        LeakyReLU(1),

        Dense(6, activation = 'softmax')
        ])
model.summary()
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass an `input_shape`/`i
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Model: "sequential"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ conv2d (Conv2D)                 │ (None, 148, 148, 16)   │           448 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ batch_normalization             │ (None, 148, 148, 16)   │            64 │
│ (BatchNormalization)            │                        │               │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ leaky_re_lu (LeakyReLU)         │ (None, 148, 148, 16)   │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ conv2d_1 (Conv2D)               │ (None, 146, 146, 32)   │         4,640 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ batch_normalization_1           │ (None, 146, 146, 32)   │           128 │
│ (BatchNormalization)            │                        │               │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ leaky_re_lu_1 (LeakyReLU)       │ (None, 146, 146, 32)   │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ max_pooling2d (MaxPooling2D)    │ (None, 29, 29, 32)     │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ conv2d_2 (Conv2D)               │ (None, 27, 27, 64)     │        18,496 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ batch_normalization_2           │ (None, 27, 27, 64)     │           256 │
│ (BatchNormalization)            │                        │               │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ leaky_re_lu_2 (LeakyReLU)       │ (None, 27, 27, 64)     │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ conv2d_3 (Conv2D)               │ (None, 25, 25, 128)    │        73,856 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ batch_normalization_3           │ (None, 25, 25, 128)    │           512 │
│ (BatchNormalization)            │                        │               │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ leaky_re_lu_3 (LeakyReLU)       │ (None, 25, 25, 128)    │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ max_pooling2d_1 (MaxPooling2D)  │ (None, 5, 5, 128)      │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ flatten (Flatten)               │ (None, 3200)           │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense (Dense)                   │ (None, 64)             │       204,864 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dropout (Dropout)               │ (None, 64)             │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ batch_normalization_4           │ (None, 64)             │           256 │
│ (BatchNormalization)            │                        │               │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ leaky_re_lu_4 (LeakyReLU)       │ (None, 64)             │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_1 (Dense)                 │ (None, 32)             │         2,080 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dropout_1 (Dropout)             │ (None, 32)             │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ batch_normalization_5           │ (None, 32)             │           128 │
│ (BatchNormalization)            │                        │               │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ leaky_re_lu_5 (LeakyReLU)       │ (None, 32)             │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_2 (Dense)                 │ (None, 16)             │           528 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dropout_2 (Dropout)             │ (None, 16)             │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ batch_normalization_6           │ (None, 16)             │            64 │
│ (BatchNormalization)            │                        │               │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ leaky_re_lu_6 (LeakyReLU)       │ (None, 16)             │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_3 (Dense)                 │ (None, 6)              │           102 │
└─────────────────────────────────┴────────────────────────┴───────────────┘
 Total params: 306,422 (1.17 MB)
 Trainable params: 305,718 (1.17 MB)
 Non-trainable params: 704 (2.75 KB)
```

```python
model.compile(loss = 'categorical_crossentropy', optimizer = Adam(0.0005),metrics=['accuracy'])
```

```python
epochs = 70
batch_size = 128
history = model.fit(x_train, y_train, batch_size = batch_size, epochs = epochs, validation_data = (x_val, y_val))
```

```
Epoch 37/70
```

```
15/15 ━━━━━━━━━━━━━━━━━━━━ 3s 108ms/step - accuracy: 0.9878 - loss: 0.0894 - val_accuracy: 0.7521 - val_loss: 0.7028
Epoch 38/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 121ms/step - accuracy: 0.9957 - loss: 0.0723 - val_accuracy: 0.7521 - val_loss: 0.7143
Epoch 39/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 112ms/step - accuracy: 0.9958 - loss: 0.0718 - val_accuracy: 0.6938 - val_loss: 0.9068
Epoch 40/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 109ms/step - accuracy: 0.9982 - loss: 0.0635 - val_accuracy: 0.6792 - val_loss: 0.9688
Epoch 41/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 109ms/step - accuracy: 0.9967 - loss: 0.0645 - val_accuracy: 0.7458 - val_loss: 0.7778
Epoch 42/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 119ms/step - accuracy: 0.9998 - loss: 0.0555 - val_accuracy: 0.7188 - val_loss: 0.8426
Epoch 43/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 120ms/step - accuracy: 0.9975 - loss: 0.0579 - val_accuracy: 0.7167 - val_loss: 0.8081
Epoch 44/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 112ms/step - accuracy: 0.9944 - loss: 0.0634 - val_accuracy: 0.7417 - val_loss: 0.8041
Epoch 45/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 3s 118ms/step - accuracy: 0.9973 - loss: 0.0539 - val_accuracy: 0.7063 - val_loss: 0.7947
Epoch 46/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 134ms/step - accuracy: 0.9991 - loss: 0.0539 - val_accuracy: 0.7125 - val_loss: 0.8224
Epoch 47/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 109ms/step - accuracy: 0.9966 - loss: 0.0605 - val_accuracy: 0.7208 - val_loss: 0.8895
Epoch 48/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 109ms/step - accuracy: 0.9967 - loss: 0.0549 - val_accuracy: 0.7583 - val_loss: 0.8126
Epoch 49/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 110ms/step - accuracy: 0.9965 - loss: 0.0589 - val_accuracy: 0.7688 - val_loss: 0.6969
Epoch 50/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 110ms/step - accuracy: 0.9983 - loss: 0.0525 - val_accuracy: 0.6833 - val_loss: 1.0243
Epoch 51/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 3s 116ms/step - accuracy: 0.9946 - loss: 0.0571 - val_accuracy: 0.7083 - val_loss: 0.9335
Epoch 52/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 124ms/step - accuracy: 0.9967 - loss: 0.0543 - val_accuracy: 0.7250 - val_loss: 0.9021
Epoch 53/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 109ms/step - accuracy: 0.9982 - loss: 0.0443 - val_accuracy: 0.7417 - val_loss: 0.8129
Epoch 54/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 109ms/step - accuracy: 0.9970 - loss: 0.0381 - val_accuracy: 0.7604 - val_loss: 0.7317
Epoch 55/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 108ms/step - accuracy: 0.9984 - loss: 0.0433 - val_accuracy: 0.7437 - val_loss: 0.7507
Epoch 56/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 110ms/step - accuracy: 0.9983 - loss: 0.0399 - val_accuracy: 0.7437 - val_loss: 0.8155
Epoch 57/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 3s 109ms/step - accuracy: 0.9994 - loss: 0.0362 - val_accuracy: 0.7604 - val_loss: 0.7092
Epoch 58/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 3s 119ms/step - accuracy: 1.0000 - loss: 0.0330 - val_accuracy: 0.7604 - val_loss: 0.7135
Epoch 59/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 109ms/step - accuracy: 0.9998 - loss: 0.0287 - val_accuracy: 0.7458 - val_loss: 0.8960
Epoch 60/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 3s 112ms/step - accuracy: 0.9990 - loss: 0.0358 - val_accuracy: 0.7417 - val_loss: 0.7619
Epoch 61/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 110ms/step - accuracy: 0.9995 - loss: 0.0289 - val_accuracy: 0.7479 - val_loss: 0.7870
Epoch 62/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 109ms/step - accuracy: 0.9998 - loss: 0.0288 - val_accuracy: 0.7188 - val_loss: 0.9003
Epoch 63/70
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 109ms/step - accuracy: 1.0000 - loss: 0.0265 - val_accuracy: 0.7417 - val_loss: 0.8387
Epoch 64/70
```

```
model.save("/content/drive/MyDrive/Data science/project-13/model.keras")
```

```
plt.figure(figsize=(12, 5))
plt.plot(history.history['accuracy'], color='r')
plt.plot(history.history['val_accuracy'], color='b')
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epochs')
plt.legend(['train', 'val'])
plt.show()
```