```
!pip install kaggle
```

```
Requirement already satisfied: kaggle in /usr/local/lib/python3.11/dist-packages (1.7.4.5)
Requirement already satisfied: bleach in /usr/local/lib/python3.11/dist-packages (from kaggle) (6.2.0)
Requirement already satisfied: certifi>=14.05.14 in /usr/local/lib/python3.11/dist-packages (from kaggle) (2025.7.14)
Requirement already satisfied: charset-normalizer in /usr/local/lib/python3.11/dist-packages (from kaggle) (3.4.2)
Requirement already satisfied: idna in /usr/local/lib/python3.11/dist-packages (from kaggle) (3.10)
Requirement already satisfied: protobuf in /usr/local/lib/python3.11/dist-packages (from kaggle) (5.29.5)
Requirement already satisfied: python-dateutil>=2.5.3 in /usr/local/lib/python3.11/dist-packages (from kaggle) (2.9.0.post0)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.11/dist-packages (from kaggle) (8.0.4)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from kaggle) (2.32.3)
Requirement already satisfied: setuptools>=21.0.0 in /usr/local/lib/python3.11/dist-packages (from kaggle) (75.2.0)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.11/dist-packages (from kaggle) (1.17.0)
Requirement already satisfied: text-unidecode in /usr/local/lib/python3.11/dist-packages (from kaggle) (1.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (from kaggle) (4.67.1)
Requirement already satisfied: urllib3>=1.15.1 in /usr/local/lib/python3.11/dist-packages (from kaggle) (2.5.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.11/dist-packages (from kaggle) (0.5.1)
```

Importing Data

```
#Code from kaggle
import kagglehub

path = kagglehub.dataset_download("mostafaabla/garbage-classification")

print("Path to dataset files:", path)
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
/tmp/ipython-input-29-3222240967.py in <cell line: 0>()
      1 import kagglehub
      2
----> 3 path = kagglehub.dataset_download("mostafaabla/garbage-classification")
      4
      5 print("Path to dataset files:", path)

 9 frames
/usr/local/lib/python3.11/dist-packages/google/colab/_message.py in read_reply_from_input(message_id, timeout_sec)
     94     reply = _read_next_input_message()
     95     if reply == _NOT_READY or not isinstance(reply, dict):
---> 96       time.sleep(0.025)
     97       continue
     98     if (

KeyboardInterrupt:
```

```
Start coding or generate with AI.
```

```
import shutil
import os

# source and destination paths
source_path = '/kaggle/input/garbage-classification/garbage_classification'
destination_path = '/content/garbage_classification_writable'


if not os.path.exists(destination_path):
    os.makedirs(destination_path)

# the dataset
print(f"Copying data from {source_path} to {destination_path}...")
shutil.copytree(source_path, destination_path, dirs_exist_ok=True)
print("Copy complete.")
```

```
Copying data from /kaggle/input/garbage-classification/garbage_classification to /content/garbage_classification_writable...
Copy complete.
```

Cleaning data

```
!pip install imagehash opencv-python-headless
```

```
Collecting imagehash
  Downloading ImageHash-4.3.2-py2.py3-none-any.whl.metadata (8.4 kB)
Requirement already satisfied: opencv-python-headless in /usr/local/lib/python3.11/dist-packages (4.12.0.88)
Requirement already satisfied: PyWavelets in /usr/local/lib/python3.11/dist-packages (from imagehash) (1.8.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (from imagehash) (2.0.2)
Requirement already satisfied: pillow in /usr/local/lib/python3.11/dist-packages (from imagehash) (11.3.0)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from imagehash) (1.16.0)
Downloading ImageHash-4.3.2-py2.py3-none-any.whl (296 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 296.7/296.7 kB 5.9 MB/s eta 0:00:00
Installing collected packages: imagehash
Successfully installed imagehash-4.3.2
```

```
from pathlib import Path
import os, shutil, hashlib, json, csv, random
from PIL import Image, UnidentifiedImageError
import numpy as np
import pandas as pd
from tqdm import tqdm
import imagehash
import cv2
```

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

```
!pip install tensorflow pillow opencv-python imagehash pandas tqdm matplotlib seaborn scikit-learn
```

```
# Import libraries
import os
from pathlib import Path
import pandas as pd
import numpy as np
from PIL import Image, UnidentifiedImageError
import matplotlib.pyplot as plt
import seaborn as sns
from tqdm import tqdm
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.11/dist-packages (2.18.0)
Requirement already satisfied: pillow in /usr/local/lib/python3.11/dist-packages (11.3.0)
Requirement already satisfied: opencv-python in /usr/local/lib/python3.11/dist-packages (4.12.0.88)
Requirement already satisfied: imagehash in /usr/local/lib/python3.11/dist-packages (4.3.2)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (4.67.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: seaborn in /usr/local/lib/python3.11/dist-packages (0.13.2)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (1.6.1)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (25.2.10)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.6.0
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (18.1.1)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.4.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from tensorflow) (25.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.3 in /usr/local/lib/pyth
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2.32.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-packages (from tensorflow) (75.2.0)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.17.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.1.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (4.14.1)
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.17.2)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.73.1)
Requirement already satisfied: tensorboard<2.19,>=2.18 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2.18.0)
Requirement already satisfied: keras>=3.5.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.8.0)
Requirement already satisfied: numpy<2.1.0,>=1.26.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2.0.2)
Requirement already satisfied: h5py>=3.11.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.14.0)
Requirement already satisfied: ml-dtypes<0.5.0,>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.4.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.37
Requirement already satisfied: PyWavelets in /usr/local/lib/python3.11/dist-packages (from imagehash) (1.8.0)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from imagehash) (1.16.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.59.0)
```

```
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.2.3)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.5.1)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (3.6.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.11/dist-packages (from astunparse>=1.6.0->tensorflow) (0.4
Requirement already satisfied: rich in /usr/local/lib/python3.11/dist-packages (from keras>=3.5.0->tensorflow) (13.9.4)
Requirement already satisfied: namex in /usr/local/lib/python3.11/dist-packages (from keras>=3.5.0->tensorflow) (0.1.0)
Requirement already satisfied: optree in /usr/local/lib/python3.11/dist-packages (from keras>=3.5.0->tensorflow) (0.16.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.21.0->tensorfl
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.21.0->tensorflow) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.21.0->tensorflow) (2
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.21.0->tensorflow) (2
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.11/dist-packages (from tensorboard<2.19,>=2.18->tensorflow) (
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.11/dist-packages (from tensorboard<2.19
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from tensorboard<2.19,>=2.18->tensorflow) (
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.11/dist-packages (from werkzeug>=1.0.1->tensorboard<2.19,>=
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich->keras>=3.5.0->tensorflow)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich->keras>=3.5.0->tensorflow
```

```
data_path = Path('/content/garbage_classification_writable')
print("Class folders:", os.listdir(data_path))
```

```
Class folders: ['paper', 'trash', 'cardboard', 'plastic', 'clothes', 'metal', 'shoes', 'battery', 'green-glass', 'brown-glass', 'biologi
```

```
#corrupt image removal
def is_valid_image(img_path):
    try:
        with Image.open(img_path) as img:
            img.verify()
        return True
    except:
        return False

valid_images = []
for folder in data_path.iterdir():
    if folder.is_dir():
        for img in folder.iterdir():
            if is_valid_image(img):
                valid_images.append(img)
```

```
from PIL import Image

def preprocess_and_save(img_path, target_size=(224,224)):
    with Image.open(img_path) as img:
        img = img.convert('RGB')
        img = img.resize(target_size)
        img.save(img_path)

for folder in data_path.iterdir():
    if folder.is_dir():
        for img in tqdm(folder.iterdir(), desc=f"Resizing {folder.name}"):
            preprocess_and_save(img)
```

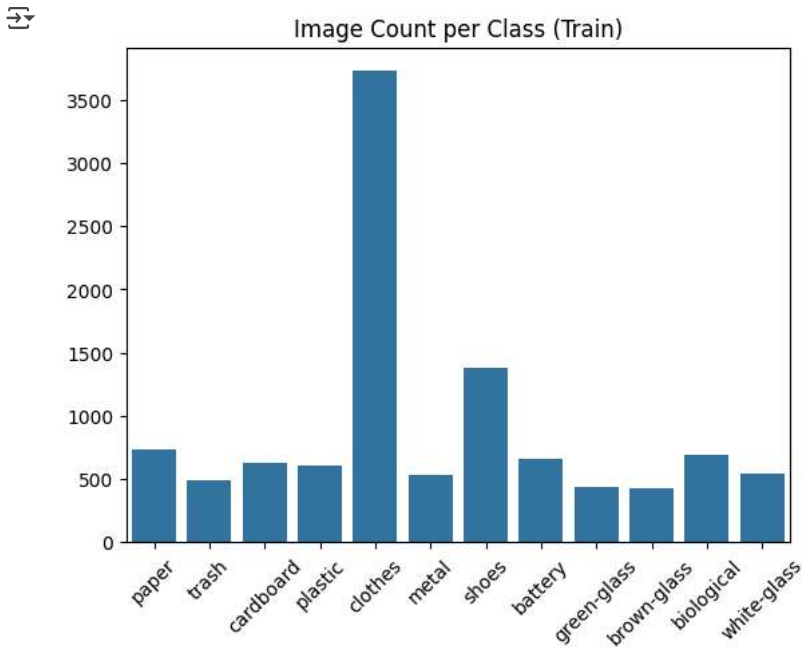
```
Resizing paper: 1050it [00:04, 261.82it/s]
Resizing trash: 697it [00:01, 535.65it/s]
Resizing cardboard: 891it [00:03, 251.00it/s]
Resizing plastic: 865it [00:02, 313.89it/s]
Resizing clothes: 5325it [00:28, 189.26it/s]
Resizing metal: 769it [00:02, 340.42it/s]
Resizing shoes: 1977it [00:05, 362.92it/s]
Resizing battery: 945it [00:02, 376.49it/s]
Resizing green-glass: 629it [00:01, 454.50it/s]
Resizing brown-glass: 607it [00:01, 444.75it/s]
Resizing biological: 985it [00:01, 512.04it/s]
Resizing white-glass: 775it [00:02, 379.45it/s]
```

```
import shutil, random

train_dir = Path('/content/data/train')
val_dir = Path('/content/data/val')
test_dir = Path('/content/data/test')

for split_dir in [train_dir, val_dir, test_dir]:
    split_dir.mkdir(parents=True, exist_ok=True)

split_ratio = {'train': 0.7, 'val': 0.15, 'test': 0.15}
random.seed(42)
```

```
for folder in data_path.iterdir():
    if folder.is_dir():
        images = list(folder.iterdir())
        random.shuffle(images)
        n_train = int(split_ratio['train'] * len(images))
        n_val = int(split_ratio['val'] * len(images))

        for img in images[:n_train]:
            dest = train_dir / folder.name
            dest.mkdir(exist_ok=True)
            shutil.copy(img, dest)
        for img in images[n_train:n_train+n_val]:
            dest = val_dir / folder.name
            dest.mkdir(exist_ok=True)
            shutil.copy(img, dest)
        for img in images[n_train+n_val:]:
            dest = test_dir / folder.name
            dest.mkdir(exist_ok=True)
            shutil.copy(img, dest)
```

EDA

```
import glob
from collections import Counter

train_counts = {}
for folder in train_dir.iterdir():
    train_counts[folder.name] = len(list(folder.iterdir()))

sns.barplot(x=list(train_counts.keys()), y=list(train_counts.values()))
plt.xticks(rotation=45)
plt.title("Image Count per Class (Train)")
plt.show()
```




```
import os

dataset_path = '/content/data/train'
categories = os.listdir(dataset_path)
print("Classes found:", categories)
```

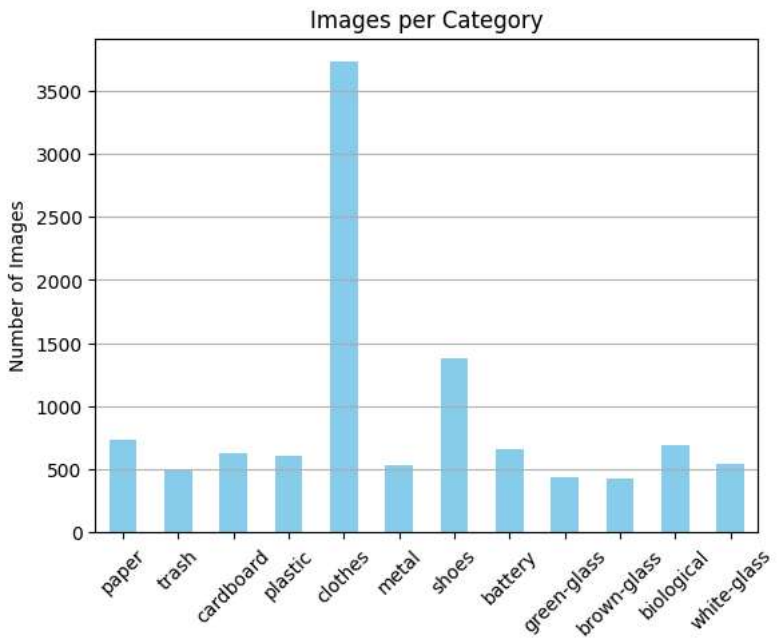
```
Classes found: ['paper', 'trash', 'cardboard', 'plastic', 'clothes', 'metal', 'shoes', 'battery', 'green-glass', 'brown-glass', 'biologi
```

```
class_counts = {cls: len(os.listdir(os.path.join(dataset_path, cls))) for cls in categories}
print("Image counts per class:")
```

```
for cls, count in class_counts.items():
    print(f"{cls}: {count}")
```

```
Image counts per class:
paper: 735
trash: 487
cardboard: 623
plastic: 605
clothes: 3727
metal: 538
shoes: 1383
battery: 661
green-glass: 440
brown-glass: 424
biological: 689
white-glass: 542
```

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.DataFrame.from_dict(class_counts, orient='index', columns=['Count'])
df.plot(kind='bar', legend=False, title='Images per Category', rot=45, color='skyblue')
plt.ylabel('Number of Images')
plt.grid(axis='y')
plt.show()
```



MODEL TRAINING

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

IMG_SIZE = (224, 224)
BATCH_SIZE = 32

datagen_train = ImageDataGenerator(rescale=1./255, horizontal_flip=True, rotation_range=15)
datagen_val = ImageDataGenerator(rescale=1./255)

dtrain = datagen_train.flow_from_directory(train_dir, target_size=IMG_SIZE, batch_size=BATCH_SIZE, class_mode='categorical')
dval = datagen_val.flow_from_directory(val_dir, target_size=IMG_SIZE, batch_size=BATCH_SIZE, class_mode='categorical')
```

```
Found 10854 images belonging to 12 classes.
Found 2321 images belonging to 12 classes.
```

```
from tensorflow.keras import layers, models

model_cnn = models.Sequential([
    layers.Conv2D(32, (3,3), activation='relu', input_shape=(224,224,3)),
```

```
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(64, (3,3), activation='relu'),
    layers.MaxPooling2D((2,2)),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(12, activation='softmax')
])

model_cnn.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
history_cnn = model_cnn.fit(dtrain, validation_data=dval, epochs=10)
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass an `input_shape`/`inpu
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset` class
  self._warn_if_super_not_called()
Epoch 1/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 138s 384ms/step - accuracy: 0.4024 - loss: 2.9229 - val_accuracy: 0.5773 - val_loss: 1.2934
Epoch 2/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 134s 377ms/step - accuracy: 0.6227 - loss: 1.1541 - val_accuracy: 0.6665 - val_loss: 1.0508
Epoch 3/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 130s 381ms/step - accuracy: 0.6799 - loss: 0.9656 - val_accuracy: 0.6678 - val_loss: 1.0320
Epoch 4/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 128s 376ms/step - accuracy: 0.7272 - loss: 0.8285 - val_accuracy: 0.6855 - val_loss: 0.9805
Epoch 5/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 128s 375ms/step - accuracy: 0.7540 - loss: 0.7567 - val_accuracy: 0.7169 - val_loss: 0.9383
Epoch 6/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 143s 378ms/step - accuracy: 0.7740 - loss: 0.6707 - val_accuracy: 0.7152 - val_loss: 0.9371
Epoch 7/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 132s 389ms/step - accuracy: 0.7847 - loss: 0.6584 - val_accuracy: 0.7505 - val_loss: 0.8590
Epoch 8/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 127s 374ms/step - accuracy: 0.8136 - loss: 0.5890 - val_accuracy: 0.7411 - val_loss: 0.8668
Epoch 9/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 128s 377ms/step - accuracy: 0.8301 - loss: 0.5137 - val_accuracy: 0.7251 - val_loss: 1.0103
Epoch 10/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 141s 375ms/step - accuracy: 0.8373 - loss: 0.4821 - val_accuracy: 0.7385 - val_loss: 0.9199
```

```
from tensorflow.keras.applications import MobileNetV2
base_model = MobileNetV2(weights='imagenet', include_top=False, input_shape=(224,224,3))
base_model.trainable = False

model_mobilenet = models.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.Dense(128, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(12, activation='softmax')
])

model_mobilenet.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
history_mobilenet = model_mobilenet.fit(dtrain, validation_data=dval, epochs=10)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mobilenet_v2/mobilenet_v2_weights_tf_dim_ordering_tf_
9406464/9406464 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
Epoch 1/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 147s 403ms/step - accuracy: 0.7414 - loss: 0.8281 - val_accuracy: 0.9156 - val_loss: 0.2664
Epoch 2/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 128s 375ms/step - accuracy: 0.9140 - loss: 0.2782 - val_accuracy: 0.9255 - val_loss: 0.2289
Epoch 3/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 126s 371ms/step - accuracy: 0.9288 - loss: 0.2163 - val_accuracy: 0.9328 - val_loss: 0.2195
Epoch 4/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 126s 371ms/step - accuracy: 0.9445 - loss: 0.1653 - val_accuracy: 0.9302 - val_loss: 0.2349
Epoch 5/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 127s 373ms/step - accuracy: 0.9508 - loss: 0.1448 - val_accuracy: 0.9319 - val_loss: 0.2048
Epoch 6/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 129s 378ms/step - accuracy: 0.9533 - loss: 0.1461 - val_accuracy: 0.9427 - val_loss: 0.2047
Epoch 7/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 128s 378ms/step - accuracy: 0.9611 - loss: 0.1125 - val_accuracy: 0.9349 - val_loss: 0.2024
Epoch 8/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 125s 367ms/step - accuracy: 0.9641 - loss: 0.1065 - val_accuracy: 0.9375 - val_loss: 0.2100
Epoch 9/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 127s 373ms/step - accuracy: 0.9661 - loss: 0.0991 - val_accuracy: 0.9410 - val_loss: 0.2200
Epoch 10/10
340/340 ━━━━━━━━━━━━━━━━━━━━ 124s 365ms/step - accuracy: 0.9675 - loss: 0.0909 - val_accuracy: 0.9405 - val_loss: 0.2168
```

Evaluation

```
from sklearn.metrics import classification_report, confusion_matrix
import numpy as np

dtest = datagen_val.flow_from_directory(test_dir, target_size=IMG_SIZE, batch_size=BATCH_SIZE, class_mode='categorical', shuffle=False)

preds = model_mobilenet.predict(dtest)
y_pred = np.argmax(preds, axis=1)
print(classification_report(dtest.classes, y_pred, target_names=dtest.class_indices.keys()))
```

```
Found 2340 images belonging to 12 classes.
74/74 ━━━━━━━━━━━━━━━━━━━━ 10s 92ms/step
              precision    recall  f1-score   support

     battery       0.98      0.92      0.95       143
  biological       0.97      0.96      0.96       149
 brown-glass       0.86      0.87      0.86        92
   cardboard       0.88      0.93      0.91       135
     clothes       0.98      0.98      0.98       800
 green-glass       0.89      0.89      0.89        95
       metal       0.74      0.86      0.80       116
       paper       0.92      0.85      0.88       158
     plastic       0.82      0.82      0.82       131
       shoes       0.95      0.96      0.96       298
       trash       0.93      0.91      0.92       106
 white-glass       0.83      0.74      0.78       117

    accuracy                           0.92      2340
   macro avg       0.90      0.89      0.89      2340
weighted avg       0.92      0.92      0.92      2340
```

Inference and saving model

```
model_mobilenet.save('/content/models/best_model.keras')
```

```
from tensorflow.keras.models import load_model
import numpy as np
from tensorflow.keras.preprocessing import image

model = load_model('/content/models/best_model.keras')

img_path = '/content/data/test/battery/battery101.jpg'
img = image.load_img(img_path, target_size=IMG_SIZE)
img_array = image.img_to_array(img) / 255.0
img_array = np.expand_dims(img_array, axis=0)

pred = model.predict(img_array)
class_idx = np.argmax(pred, axis=1)[0]
print("Predicted class:", list(dtrain.class_indices.keys())[class_idx])
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 5s 5s/step
Predicted class: battery
```