Dog Breed Identification Using Transfer Learning

1) Install Required Libraries

```
pip install tensorflow numpy matplotlib
```

2) Import Libraries

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras import layers, models
import matplotlib.pyplot as plt
```

3) Dataset Structure

```
dataset/
    train/
        breed1/
        breed2/
        breed3/
    validation/
        breed1/
        breed2/
        breed3/
```

```
img_size = 224
batch_size = 32

train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    zoom_range=0.2,
    horizontal_flip=True
)

val_datagen = ImageDataGenerator(rescale=1./255)

train_data = train_datagen.flow_from_directory(
    "dataset/train",
    target_size=(img_size, img_size),
    batch_size=batch_size,
    class_mode="categorical"
)

val_data = val_datagen.flow_from_directory(
    "dataset/validation",
    target_size=(img_size, img_size),
    batch_size=batch_size,
    class_mode="categorical"
)
```

4) Load Pretrained Model

```
base_model = MobileNetV2(
    weights="imagenet",
    include_top=False,
    input_shape=(224, 224, 3)
)

base_model.trainable = False
```

5) Add Custom Layers

```
model = models.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.Dense(128, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(train_data.num_classes, activation='softmax')
])
```

6) Compile Model

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

7) Train Model

```
history = model.fit(
    train_data,
    validation_data=val_data,
    epochs=10
)
```

8) Save Model

```
model.save("dog_breed_model.h5")
```

9) Predict New Image

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

img_path = "test.jpg"
img = image.load_img(img_path, target_size=(224, 224))
img_array = image.img_to_array(img) / 255.0
img_array = np.expand_dims(img_array, axis=0)

prediction = model.predict(img_array)
class_index = np.argmax(prediction)
class_labels = list(train_data.class_indices.keys())

print("Predicted Breed:", class_labels[class_index])
```