Deep Residual Learning for Image Recognition

Project Members:

- Aditya Pradeep Waghmode - Z23737910
- Surendra Ramisetty - Z23734231
- Srilatha Jidugu - Z23747421

Trail 1: Code

Google Colab link for trail 1 code:

https://colab.research.google.com/drive/1fKeMZVo7Rw2LL6zvBPTy9GEzH1mzPXGn?usp=sharing

Import necessary libraries

```
[4] import numpy as np
    from keras.datasets import cifar10
    from keras.utils import to_categorical
    from keras.models import Sequential
    from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, Activation
    from keras.optimizers import Adam
    from keras.callbacks import ModelCheckpoint
    from keras.preprocessing.image import ImageDataGenerator
    import matplotlib.pyplot as plt
```

Load CIFAR-10 dataset and split into training and testing sets

```
[5] (x_train, y_train), (x_test, y_test) = cifar10.load_data()

    Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
    170498071/170498071 [==============================] - 4s 0us/step
```

Create a validation set by randomly selecting 20% of the training images

```
[6] validation_split = 0.2
    validation_samples = int(len(x_train) * validation_split)
    x_val = x_train[:validation_samples]
    y_val = y_train[:validation_samples]
    x_train = x_train[validation_samples:]
    y_train = y_train[validation_samples:]
```

Scale pixel values to a range between 0 and 1

```
x_train = x_train.astype('float32') / 255
x_val = x_val.astype('float32') / 255
x_test = x_test.astype('float32') / 255
```

Convert labels to binary class matrices

```
[8] y_train = to_categorical(y_train, 10)
    y_val = to_categorical(y_val, 10)
    y_test = to_categorical(y_test, 10)
```

Double-click (or enter) to edit

Import necessary libraries for model creation

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten, Activation
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from keras.preprocessing.image import ImageDataGenerator
from keras.src.layers.serialization import activation
```

Build CNN model with data augmentation

```
[10] data_generator = ImageDataGenerator(rotation_range=10, width_shift_range=0.1, height_shift_range=0.1, horizontal_flip=True)
     data_generator.fit(x_train)


     model_augmented = Sequential()
     pretraining_model = tf.keras.applications.ResNet152V2(
         include_top=False,
         weights="imagenet",
         input_tensor=None,
         input_shape=(32, 32, 3),
         pooling=None,  # No global pooling in the pre-trained model
         classes=10
     )

     for layer in pretraining_model.layers:
         layer.trainable = True
```

```
model_augmented.add(pretraining_model)
model_augmented.add(Conv2D(32, (3, 3), activation='relu', padding='same', input_shape=(32, 32, 3)))
model_augmented.add(Activation('relu'))
model_augmented.add(Conv2D(32, (3, 3), activation='relu', padding='same'))
model_augmented.add(Activation('relu'))
model_augmented.add(MaxPooling2D(pool_size=(1, 1)))
model_augmented.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
model_augmented.add(Activation('relu'))
model_augmented.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
model_augmented.add(Activation('relu'))
model_augmented.add(MaxPooling2D(pool_size=(1, 1)))
model_augmented.add(Flatten())
model_augmented.add(Dense(512, activation='relu'))
model_augmented.add(Dropout(0.5))
model_augmented.add(Dense(10, activation='softmax'))

model_augmented.compile(loss='categorical_crossentropy', optimizer=Adam(lr=0.001), metrics=['accuracy'])
checkpoint_augmented = ModelCheckpoint('best_model_augmented.h5', save_best_only=True, monitor='val_loss', mode='min', verbose=1)
history_augmented = model_augmented.fit(data_generator.flow(x_train, y_train, batch_size=512), epochs=50, validation_data=(x_val, y_val), callbacks=[checkpoint_augmented])
model_augmented.load_weights('best_model_augmented.h5')
train_loss_augmented, train_accuracy_augmented = model_augmented.evaluate(x_train, y_train)
val_loss_augmented, val_accuracy_augmented = model_augmented.evaluate(x_val, y_val)
```

Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet152v2_weights_tf_dim_ordering_tf_kernels_notop.h5
234545216/234545216 [==============================] - 1s 0us/step
WARNING:absl:`lr` is deprecated in Keras optimizer, please use `learning_rate` or use the legacy optimizer, e.g.,tf.keras.optimizers.legacy.Adam.
Epoch 1/50
79/79 [==============================] - ETA: 0s - loss: 2.2148 - accuracy: 0.1395
Epoch 1: val_loss improved from inf to 1283.32739, saving model to best_model_augmented.h5
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3079: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')`.
  saving_api.save_model(
79/79 [==============================] - 159s 654ms/step - loss: 2.2148 - accuracy: 0.1395 - val_loss: 1283.3274 - val_accuracy: 0.0974
Epoch 2/50
79/79 [==============================] - ETA: 0s - loss: 1.8325 - accuracy: 0.2298
Epoch 2: val_loss improved from 1283.32739 to 3.86596, saving model to best_model_augmented.h5
79/79 [==============================] - 59s 748ms/step - loss: 1.8325 - accuracy: 0.2298 - val_loss: 3.8660 - val_accuracy: 0.1880
Epoch 3/50
79/79 [==============================] - ETA: 0s - loss: 1.6414 - accuracy: 0.3225
Epoch 3: val_loss did not improve from 3.86596
79/79 [==============================] - 45s 564ms/step - loss: 1.6414 - accuracy: 0.3225 - val_loss: 7.4576 - val_accuracy: 0.2724
Epoch 4/50
79/79 [==============================] - ETA: 0s - loss: 1.4701 - accuracy: 0.4108
Epoch 4: val_loss did not improve from 3.86596
79/79 [==============================] - 42s 528ms/step - loss: 1.4701 - accuracy: 0.4108 - val_loss: 3.9650 - val_accuracy: 0.2928
Epoch 5/50
79/79 [==============================] - ETA: 0s - loss: 1.4853 - accuracy: 0.4238
Epoch 5: val_loss improved from 3.86596 to 2.14562, saving model to best_model_augmented.h5
79/79 [==============================] - 51s 642ms/step - loss: 1.4853 - accuracy: 0.4238 - val_loss: 2.1456 - val_accuracy: 0.4281
Epoch 6/50
79/79 [==============================] - ETA: 0s - loss: 1.4107 - accuracy: 0.4665
Epoch 6: val_loss did not improve from 2.14562
79/79 [==============================] - 45s 564ms/step - loss: 1.4107 - accuracy: 0.4665 - val_loss: 2.2325 - val_accuracy: 0.4260
Epoch 7/50
79/79 [==============================] - ETA: 0s - loss: 1.3240 - accuracy: 0.5099
Epoch 7: val_loss did not improve from 2.14562

79/79 [==============================] - 44s 551ms/step - loss: 1.3240 - accuracy: 0.5099 - val_loss: 2.4001 - val_accuracy: 0.3953
Epoch 8/50
79/79 [==============================] - ETA: 0s - loss: 1.1981 - accuracy: 0.5767
Epoch 8: val_loss improved from 2.14562 to 1.70583, saving model to best_model_augmented.h5
79/79 [==============================] - 52s 655ms/step - loss: 1.1981 - accuracy: 0.5767 - val_loss: 1.7058 - val_accuracy: 0.5855
Epoch 9/50
79/79 [==============================] - ETA: 0s - loss: 1.0740 - accuracy: 0.6270
Epoch 9: val_loss improved from 1.70583 to 1.16933, saving model to best_model_augmented.h5
79/79 [==============================] - 52s 652ms/step - loss: 1.0740 - accuracy: 0.6270 - val_loss: 1.1693 - val_accuracy: 0.6300
Epoch 10/50
79/79 [==============================] - ETA: 0s - loss: 0.9996 - accuracy: 0.6625
Epoch 10: val_loss improved from 1.16933 to 1.15764, saving model to best_model_augmented.h5
79/79 [==============================] - 52s 652ms/step - loss: 0.9996 - accuracy: 0.6625 - val_loss: 1.1576 - val_accuracy: 0.6614
Epoch 11/50
79/79 [==============================] - ETA: 0s - loss: 0.9315 - accuracy: 0.6896
Epoch 11: val_loss improved from 1.15764 to 1.04676, saving model to best_model_augmented.h5
79/79 [==============================] - 49s 625ms/step - loss: 0.9315 - accuracy: 0.6896 - val_loss: 1.0468 - val_accuracy: 0.6617
Epoch 12/50
79/79 [==============================] - ETA: 0s - loss: 0.8714 - accuracy: 0.7095
Epoch 12: val_loss improved from 1.04676 to 0.97995, saving model to best_model_augmented.h5
79/79 [==============================] - 53s 674ms/step - loss: 0.8714 - accuracy: 0.7095 - val_loss: 0.9799 - val_accuracy: 0.6934
Epoch 13/50
79/79 [==============================] - ETA: 0s - loss: 0.8218 - accuracy: 0.7238
Epoch 13: val_loss did not improve from 0.97995
79/79 [==============================] - 42s 529ms/step - loss: 0.8218 - accuracy: 0.7238 - val_loss: 1.2358 - val_accuracy: 0.6334
Epoch 14/50
79/79 [==============================] - ETA: 0s - loss: 0.7795 - accuracy: 0.7421
Epoch 14: val_loss improved from 0.97995 to 0.78946, saving model to best_model_augmented.h5
79/79 [==============================] - 55s 699ms/step - loss: 0.7795 - accuracy: 0.7421 - val_loss: 0.7895 - val_accuracy: 0.7464
Epoch 15/50
79/79 [==============================] - ETA: 0s - loss: 0.7563 - accuracy: 0.7521
Epoch 15: val_loss did not improve from 0.78946
79/79 [==============================] - 44s 556ms/step - loss: 0.7563 - accuracy: 0.7521 - val_loss: 1.0822 - val_accuracy: 0.6820
Epoch 16/50
79/79 [==============================] - ETA: 0s - loss: 0.7327 - accuracy: 0.7608
Epoch 16: val_loss improved from 0.78946 to 0.76350, saving model to best_model_augmented.h5
79/79 [==============================] - 50s 631ms/step - loss: 0.7327 - accuracy: 0.7608 - val_loss: 0.7635 - val_accuracy: 0.7566
Epoch 17/50
79/79 [==============================] - ETA: 0s - loss: 0.7046 - accuracy: 0.7702
Epoch 17: val_loss did not improve from 0.76350
79/79 [==============================] - 44s 552ms/step - loss: 0.7046 - accuracy: 0.7702 - val_loss: 0.8856 - val_accuracy: 0.7191
Epoch 18/50
79/79 [==============================] - ETA: 0s - loss: 0.6594 - accuracy: 0.7856
Epoch 18: val_loss did not improve from 0.76350

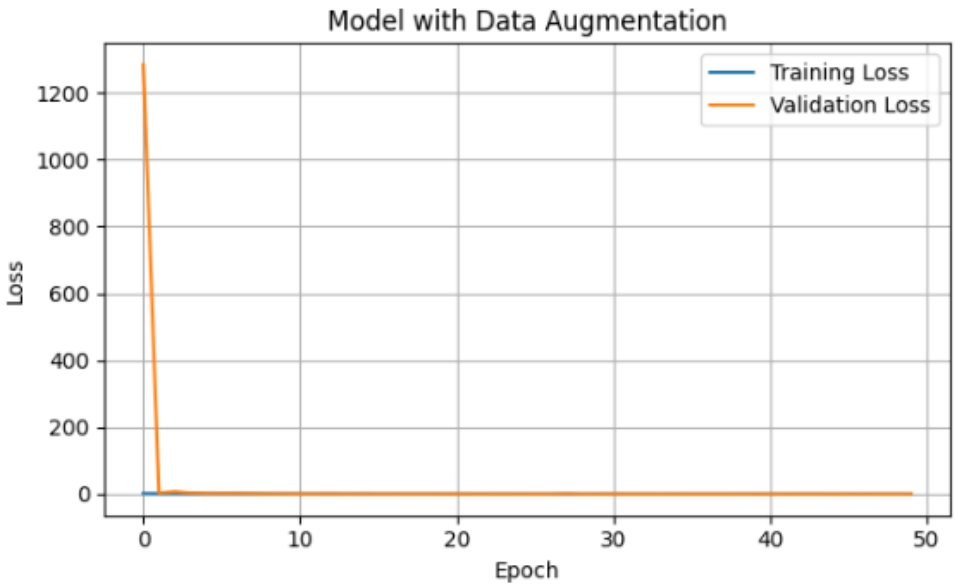
79/79 [==============================] - 45s 566ms/step - loss: 0.6594 - accuracy: 0.7856 - val_loss: 0.8845 - val_accuracy: 0.7286
Epoch 19/50
79/79 [==============================] - ETA: 0s - loss: 0.6426 - accuracy: 0.7910
Epoch 19: val_loss did not improve from 0.76350
79/79 [==============================] - 43s 542ms/step - loss: 0.6426 - accuracy: 0.7910 - val_loss: 0.9594 - val_accuracy: 0.7147
Epoch 20/50
79/79 [==============================] - ETA: 0s - loss: 0.6377 - accuracy: 0.7928
Epoch 20: val_loss did not improve from 0.76350
79/79 [==============================] - 44s 556ms/step - loss: 0.6377 - accuracy: 0.7928 - val_loss: 0.8898 - val_accuracy: 0.7321
Epoch 21/50
79/79 [==============================] - ETA: 0s - loss: 0.6053 - accuracy: 0.8020
Epoch 21: val_loss did not improve from 0.76350
79/79 [==============================] - 42s 532ms/step - loss: 0.6053 - accuracy: 0.8020 - val_loss: 0.8557 - val_accuracy: 0.7541
Epoch 22/50
79/79 [==============================] - ETA: 0s - loss: 0.5857 - accuracy: 0.8103
Epoch 22: val_loss improved from 0.76350 to 0.73701, saving model to best_model_augmented.h5
79/79 [==============================] - 55s 700ms/step - loss: 0.5857 - accuracy: 0.8103 - val_loss: 0.7370 - val_accuracy: 0.7733
Epoch 23/50
79/79 [==============================] - ETA: 0s - loss: 0.5693 - accuracy: 0.8164
Epoch 23: val_loss did not improve from 0.73701
79/79 [==============================] - 41s 521ms/step - loss: 0.5693 - accuracy: 0.8164 - val_loss: 0.7746 - val_accuracy: 0.7581
Epoch 24/50
79/79 [==============================] - ETA: 0s - loss: 0.5504 - accuracy: 0.8238
Epoch 24: val_loss did not improve from 0.73701
79/79 [==============================] - 44s 558ms/step - loss: 0.5504 - accuracy: 0.8238 - val_loss: 0.8447 - val_accuracy: 0.7334
Epoch 25/50
79/79 [==============================] - ETA: 0s - loss: 0.5465 - accuracy: 0.8206
Epoch 25: val_loss did not improve from 0.73701
79/79 [==============================] - 45s 566ms/step - loss: 0.5465 - accuracy: 0.8206 - val_loss: 0.8152 - val_accuracy: 0.7591
Epoch 26/50
79/79 [==============================] - ETA: 0s - loss: 0.5265 - accuracy: 0.8277
Epoch 26: val_loss improved from 0.73701 to 0.66414, saving model to best_model_augmented.h5
79/79 [==============================] - 53s 670ms/step - loss: 0.5265 - accuracy: 0.8277 - val_loss: 0.6641 - val_accuracy: 0.7972
Epoch 27/50
79/79 [==============================] - ETA: 0s - loss: 0.5086 - accuracy: 0.8386
Epoch 27: val_loss did not improve from 0.66414
79/79 [==============================] - 44s 552ms/step - loss: 0.5086 - accuracy: 0.8386 - val_loss: 0.7073 - val_accuracy: 0.7749
Epoch 28/50
79/79 [==============================] - ETA: 0s - loss: 0.4855 - accuracy: 0.8437
Epoch 28: val_loss did not improve from 0.66414
79/79 [==============================] - 42s 523ms/step - loss: 0.4855 - accuracy: 0.8437 - val_loss: 1.5998 - val_accuracy: 0.6233
Epoch 29/50
79/79 [==============================] - ETA: 0s - loss: 0.4977 - accuracy: 0.8379
Epoch 29: val_loss improved from 0.66414 to 0.65613, saving model to best_model_augmented.h5

79/79 [==============================] - 51s 650ms/step - loss: 0.4977 - accuracy: 0.8379 - val_loss: 0.6561 - val_accuracy: 0.7943
Epoch 30/50
79/79 [==============================] - ETA: 0s - loss: 0.4630 - accuracy: 0.8511
Epoch 30: val_loss did not improve from 0.65613
79/79 [==============================] - 43s 548ms/step - loss: 0.4630 - accuracy: 0.8511 - val_loss: 0.8926 - val_accuracy: 0.7512
Epoch 31/50
79/79 [==============================] - ETA: 0s - loss: 0.4605 - accuracy: 0.8510
Epoch 31: val_loss did not improve from 0.65613
79/79 [==============================] - 44s 550ms/step - loss: 0.4605 - accuracy: 0.8510 - val_loss: 0.9711 - val_accuracy: 0.7432
Epoch 32/50
79/79 [==============================] - ETA: 0s - loss: 0.4621 - accuracy: 0.8508
Epoch 32: val_loss did not improve from 0.65613
79/79 [==============================] - 44s 551ms/step - loss: 0.4621 - accuracy: 0.8508 - val_loss: 0.7870 - val_accuracy: 0.7800
Epoch 33/50
79/79 [==============================] - ETA: 0s - loss: 0.4504 - accuracy: 0.8537
Epoch 33: val_loss did not improve from 0.65613
79/79 [==============================] - 44s 557ms/step - loss: 0.4504 - accuracy: 0.8537 - val_loss: 0.8519 - val_accuracy: 0.7504
Epoch 34/50
79/79 [==============================] - ETA: 0s - loss: 0.4414 - accuracy: 0.8558
Epoch 34: val_loss did not improve from 0.65613
79/79 [==============================] - 44s 553ms/step - loss: 0.4414 - accuracy: 0.8558 - val_loss: 0.6652 - val_accuracy: 0.7952
Epoch 35/50
79/79 [==============================] - ETA: 0s - loss: 0.4252 - accuracy: 0.8618
Epoch 35: val_loss did not improve from 0.65613
79/79 [==============================] - 43s 536ms/step - loss: 0.4252 - accuracy: 0.8618 - val_loss: 0.7694 - val_accuracy: 0.7682
Epoch 36/50
79/79 [==============================] - ETA: 0s - loss: 0.4058 - accuracy: 0.8674
Epoch 36: val_loss did not improve from 0.65613
79/79 [==============================] - 43s 536ms/step - loss: 0.4058 - accuracy: 0.8674 - val_loss: 0.6575 - val_accuracy: 0.8081
Epoch 37/50
79/79 [==============================] - ETA: 0s - loss: 0.3974 - accuracy: 0.8726
Epoch 37: val_loss did not improve from 0.65613
79/79 [==============================] - 43s 540ms/step - loss: 0.3974 - accuracy: 0.8726 - val_loss: 0.6886 - val_accuracy: 0.7958
Epoch 38/50
79/79 [==============================] - ETA: 0s - loss: 0.3898 - accuracy: 0.8733
Epoch 38: val_loss did not improve from 0.65613
79/79 [==============================] - 42s 528ms/step - loss: 0.3898 - accuracy: 0.8733 - val_loss: 0.7051 - val_accuracy: 0.8021
Epoch 39/50
79/79 [==============================] - ETA: 0s - loss: 0.3820 - accuracy: 0.8760
Epoch 39: val_loss did not improve from 0.65613
79/79 [==============================] - 42s 530ms/step - loss: 0.3820 - accuracy: 0.8760 - val_loss: 0.7427 - val_accuracy: 0.7900
Epoch 40/50
79/79 [==============================] - ETA: 0s - loss: 0.3656 - accuracy: 0.8798
Epoch 40: val_loss did not improve from 0.65613

```
79/79 [==============================] - 44s 552ms/step - loss: 0.3656 - accuracy: 0.8798 - val_loss:
0.9921 - val_accuracy: 0.7436
Epoch 41/50
79/79 [==============================] - ETA: 0s - loss: 0.3737 - accuracy: 0.8773
Epoch 41: val_loss did not improve from 0.65613
79/79 [==============================] - 44s 553ms/step - loss: 0.3737 - accuracy: 0.8773 - val_loss:
0.8021 - val_accuracy: 0.7753
Epoch 42/50
79/79 [==============================] - ETA: 0s - loss: 0.3637 - accuracy: 0.8831
Epoch 42: val_loss did not improve from 0.65613
79/79 [==============================] - 43s 537ms/step - loss: 0.3637 - accuracy: 0.8831 - val_loss:
0.7260 - val_accuracy: 0.7912
Epoch 43/50
79/79 [==============================] - ETA: 0s - loss: 0.3668 - accuracy: 0.8810
Epoch 43: val_loss did not improve from 0.65613
79/79 [==============================] - 45s 569ms/step - loss: 0.3668 - accuracy: 0.8810 - val_loss:
0.8774 - val_accuracy: 0.7559
Epoch 44/50
79/79 [==============================] - ETA: 0s - loss: 0.3514 - accuracy: 0.8856
Epoch 44: val_loss did not improve from 0.65613
79/79 [==============================] - 43s 540ms/step - loss: 0.3514 - accuracy: 0.8856 - val_loss:
0.8530 - val_accuracy: 0.7751
Epoch 45/50
79/79 [==============================] - ETA: 0s - loss: 0.3240 - accuracy: 0.8954
Epoch 45: val_loss did not improve from 0.65613
79/79 [==============================] - 45s 571ms/step - loss: 0.3240 - accuracy: 0.8954 - val_loss:
0.8540 - val_accuracy: 0.7555
Epoch 46/50
79/79 [==============================] - ETA: 0s - loss: 0.3412 - accuracy: 0.8902
Epoch 46: val_loss did not improve from 0.65613
79/79 [==============================] - 49s 621ms/step - loss: 0.3412 - accuracy: 0.8902 - val_loss:
0.8199 - val_accuracy: 0.7790
Epoch 47/50
79/79 [==============================] - ETA: 0s - loss: 0.3235 - accuracy: 0.8954
Epoch 47: val_loss did not improve from 0.65613
79/79 [==============================] - 49s 623ms/step - loss: 0.3235 - accuracy: 0.8954 - val_loss:
0.8914 - val_accuracy: 0.7639
Epoch 48/50
79/79 [==============================] - ETA: 0s - loss: 0.3172 - accuracy: 0.8957
Epoch 48: val_loss did not improve from 0.65613
79/79 [==============================] - 52s 657ms/step - loss: 0.3172 - accuracy: 0.8957 - val_loss:
0.7707 - val_accuracy: 0.7954
Epoch 49/50
79/79 [==============================] - ETA: 0s - loss: 0.3117 - accuracy: 0.8976
Epoch 49: val_loss did not improve from 0.65613
79/79 [==============================] - 44s 562ms/step - loss: 0.3117 - accuracy: 0.8976 - val_loss:
0.8442 - val_accuracy: 0.7903
Epoch 50/50
79/79 [==============================] - ETA: 0s - loss: 0.3104 - accuracy: 0.8993
Epoch 50: val_loss improved from 0.65613 to 0.65273, saving model to best_model_augmented.h5
79/79 [==============================] - 51s 645ms/step - loss: 0.3104 - accuracy: 0.8993 - val_loss:
0.6527 - val_accuracy: 0.8107
1250/1250 [==============================] - 42s 33ms/step - loss: 0.3486 - accuracy: 0.8863
313/313 [==============================] - 10s 31ms/step - loss: 0.6527 - accuracy: 0.8107
```

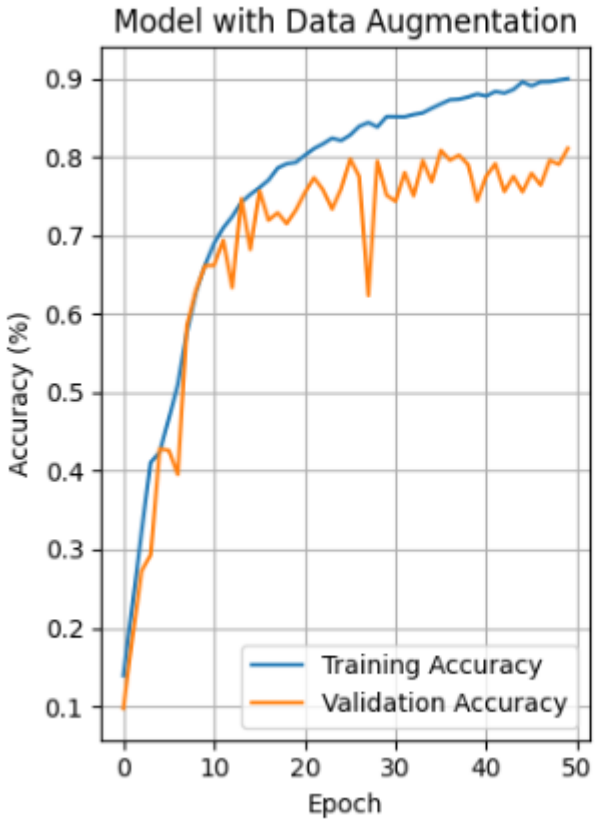
Plot: Loss vs Epoch (with data augmentation)

```
plt.figure(figsize=(12, 4))

# Plotting Training and Validation Loss
plt.subplot(1, 2, 1)
plt.plot(history_augmented.history['loss'], label='Training Loss', color='blue', marker='o')
plt.plot(history_augmented.history['val_loss'], label='Validation Loss', color='orange', marker='o')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Model with Data Augmentation')
plt.legend()
plt.grid(True)
plt.tight_layout()

plt.axhline(0, color='black',linewidth=0.5)
plt.axvline(0, color='black',linewidth=0.5)
plt.title('Training and Validation Loss Over Epochs')
plt.legend()
plt.grid(True, linestyle='--', alpha=0.7)
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.tight_layout()

plt.show()
```



Plot: Accuracy vs Epoch (with data augmentation)

```
plt.figure(figsize=(12, 4))

# Plotting Training and Validation Accuracy
plt.subplot(1, 2, 2)
plt.plot(history_augmented.history['accuracy'], label='Training Accuracy', color='green', marker='o')
plt.plot(history_augmented.history['val_accuracy'], label='Validation Accuracy', color='red', marker='o')
plt.xlabel('Epoch')
plt.ylabel('Accuracy (%)')
plt.title('Model with Data Augmentation')
plt.legend()
plt.grid(True)

plt.axhline(0, color='black', linewidth=0.5)
plt.axvline(0, color='black', linewidth=0.5)
plt.title('Training and Validation Accuracy Over Epochs')
plt.legend()
plt.grid(True, linestyle='--', alpha=0.7)
plt.xlabel('Epoch')
plt.ylabel('Accuracy (%)')
plt.legend()
plt.tight_layout()

plt.show()
```



Print training and validation accuracy and loss for the model (with data augmentation)

```
[13] print("\nModel with Data Augmentation:")
     print("Training Loss: {:.4f}".format(history_augmented.history['loss'][-1]))
     print("Training Accuracy: {:.4f}%".format(history_augmented.history['accuracy'][-1] * 100))
     print("Validation Loss: {:.4f}".format(history_augmented.history['val_loss'][-1]))
     print("Validation Accuracy: {:.4f}%".format(history_augmented.history['val_accuracy'][-1] * 100))
```

```
Model with Data Augmentation:
Training Loss: 0.3104
Training Accuracy: 89.9325%
Validation Loss: 0.6527
Validation Accuracy: 81.0700%
```

Print summary of the model (with Data Augmentation)

```
print("\nModel Summary (with Data Augmentation):")
model_augmented.summary()
```

```
Model Summary (with Data Augmentation):
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet152v2 (Functional)    (None, 1, 1, 2048)        58331648

 conv2d (Conv2D)             (None, 1, 1, 32)          589856

 activation (Activation)     (None, 1, 1, 32)          0

 conv2d_1 (Conv2D)           (None, 1, 1, 32)          9248

 activation_1 (Activation)   (None, 1, 1, 32)          0

 max_pooling2d_3 (MaxPoolin  (None, 1, 1, 32)          0
 g2D)

 conv2d_2 (Conv2D)           (None, 1, 1, 64)          18496

 activation_2 (Activation)   (None, 1, 1, 64)          0

 conv2d_3 (Conv2D)           (None, 1, 1, 64)          36928

 activation_3 (Activation)   (None, 1, 1, 64)          0

 max_pooling2d_4 (MaxPoolin  (None, 1, 1, 64)          0
 g2D)

 flatten (Flatten)           (None, 64)                0
```

```
 dense (Dense)               (None, 512)               33280

 dropout (Dropout)           (None, 512)               0

 dense_1 (Dense)             (None, 10)                5130

=================================================================
Total params: 59024586 (225.16 MB)
Trainable params: 58880842 (224.61 MB)
Non-trainable params: 143744 (561.50 KB)
```

Evaluate the model with test data

```
[15] test_loss, test_accuracy = model_augmented.evaluate(x_test, y_test)
     print(f'Test Loss: {test_loss:.4f}, Test Accuracy: {test_accuracy*100:.2f}%')

     313/313 [==============================] - 9s 29ms/step - loss: 0.7024 - accuracy: 0.8027
     Test Loss: 0.7024, Test Accuracy: 80.27%
```

Plotting the confusion matrix as heatmap

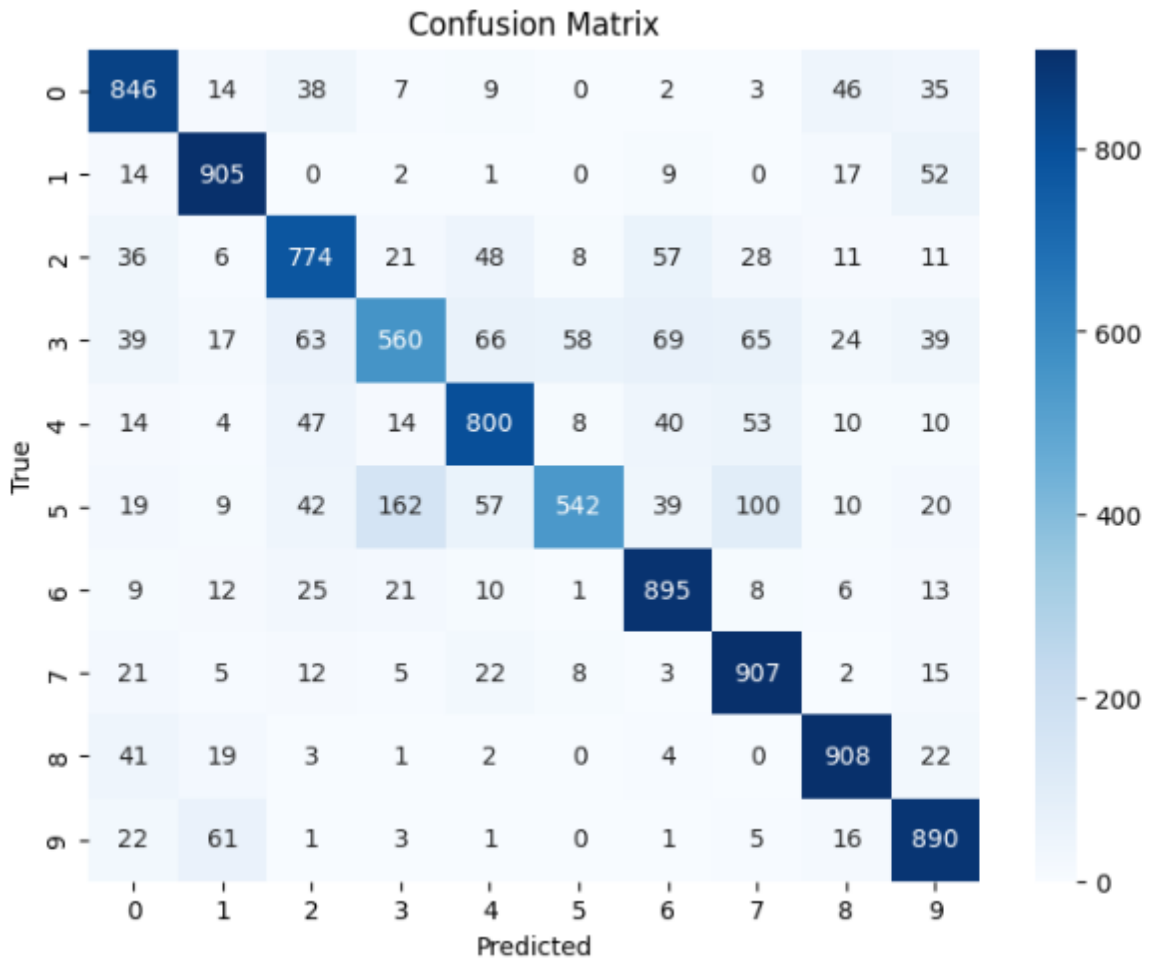
```
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score

# Predict classes for test set
y_pred = model_augmented.predict(x_test)
y_pred_classes = np.argmax(y_pred, axis=1)
y_true = np.argmax(y_test, axis=1)

# Compute confusion matrix
conf_matrix = confusion_matrix(y_true, y_pred_classes)
print("Confusion Matrix:")
print(conf_matrix)

# Compute classification report
class_report = classification_report(y_true, y_pred_classes, target_names=[str(i) for i in range(10)])
print("Classification Report:")
print(class_report)

# Compute accuracy
accuracy = accuracy_score(y_true, y_pred_classes)
print("Accuracy:", accuracy)
```

```
313/313 [==============================] - 8s 26ms/step
Confusion Matrix:
[[846  14  38   7   9   0   2   3  46  35]
 [ 14 905   0   2   1   0   9   0  17  52]
 [ 36   6 774  21  48   8  57  28  11  11]
 [ 39  17  63 560  66  58  69  65  24  39]
 [ 14   4  47  14 800   8  40  53  10  10]
 [ 19   9  42 162  57 542  39 100  10  20]
 [  9  12  25  21  10   1 895   8   6  13]
 [ 21   5  12   5  22   8   3 907   2  15]
 [ 41  19   3   1   2   0   4   0 908  22]
 [ 22  61   1   3   1   0   1   5  16 890]]
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.85      0.82      1000
           1       0.86      0.91      0.88      1000
           2       0.77      0.77      0.77      1000
           3       0.70      0.56      0.62      1000
           4       0.79      0.80      0.79      1000
           5       0.87      0.54      0.67      1000
           6       0.80      0.90      0.84      1000
           7       0.78      0.91      0.84      1000
           8       0.86      0.91      0.89      1000
           9       0.80      0.89      0.84      1000

    accuracy                           0.80     10000
   macro avg       0.80      0.80      0.80     10000
weighted avg       0.80      0.80      0.80     10000

Accuracy: 0.8027
```

Confusion matrix plot

```
[20] import seaborn as sns
     import matplotlib.pyplot as plt

     # Compute confusion matrix
     conf_matrix = confusion_matrix(y_true, y_pred_classes)

     # Plot confusion matrix heatmap
     plt.figure(figsize=(8, 6))
     sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=[str(i) for i in range(10)], yticklabels=[str(i) for i in range(10)])
     plt.title('Confusion Matrix')
     plt.xlabel('Predicted')
     plt.ylabel('True')
     plt.show()
```



Note: Due to technical issue in google colab output was not visible. So, needed to rerun the code for trail 1.

Trail 2: Code

Google Colab link for trail 1 code:

https://colab.research.google.com/drive/1UzXngwYTycR31BoRGX478JfiOddh_cGf?usp=sharing

```
[ ]  # Import necessary libraries
     import numpy as np
     from keras.datasets import cifar10
     from keras.utils import to_categorical
     from keras.models import Sequential
     from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, Activation
     from keras.optimizers import Adam
     from keras.callbacks import ModelCheckpoint
     from keras.preprocessing.image import ImageDataGenerator
     import matplotlib.pyplot as plt

     # Load CIFAR-10 dataset and split into training and testing sets
     (x_train, y_train), (x_test, y_test) = cifar10.load_data()

     # Create a validation set by randomly selecting 20% of the training images
     validation_split = 0.2
     validation_samples = int(len(x_train) * validation_split)
     x_val = x_train[:validation_samples]
     y_val = y_train[:validation_samples]
     x_train = x_train[validation_samples:]
     y_train = y_train[validation_samples:]

     # Scale pixel values to a range between 0 and 1
     x_train = x_train.astype('float32') / 255
     x_val = x_val.astype('float32') / 255
     x_test = x_test.astype('float32') / 255

     # Convert labels to binary class matrices
     y_train = to_categorical(y_train, 10)
     y_val = to_categorical(y_val, 10)
     y_test = to_categorical(y_test, 10)
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 [==============================] - 6s 0us/step
```

```python
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten, Activation
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from keras.preprocessing.image import ImageDataGenerator
from keras.src.layers.serialization import activation

# Build CNN model with data augmentation
data_generator = ImageDataGenerator(rotation_range=10, width_shift_range=0.1, height_shift_range=0.1, horizontal_flip=True)
data_generator.fit(x_train)


model_augmented = Sequential()
pretraining_model = tf.keras.applications.ResNet152V2(
    include_top=False,
    weights="imagenet",
    input_tensor=None,
    input_shape=(32, 32, 3),
    pooling=None,  # No global pooling in the pre-trained model
    classes=10
)

for layer in pretraining_model.layers:
    layer.trainable = True
```

```python
model_augmented.add(pretraining_model)
model_augmented.add(Conv2D(32, (3, 3), activation='relu', padding='same', input_shape=(32, 32, 3)))
model_augmented.add(Activation('relu'))
model_augmented.add(Conv2D(32, (3, 3), activation='relu', padding='same'))
model_augmented.add(Activation('relu'))
model_augmented.add(MaxPooling2D(pool_size=(1, 1)))
model_augmented.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
model_augmented.add(Activation('relu'))
model_augmented.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
model_augmented.add(Activation('relu'))
model_augmented.add(MaxPooling2D(pool_size=(1, 1)))
model_augmented.add(Flatten())
model_augmented.add(Dense(512, activation='relu'))
model_augmented.add(Dropout(0.5))
model_augmented.add(Dense(10, activation='softmax'))

model_augmented.compile(loss='categorical_crossentropy', optimizer=Adam(lr=0.001), metrics=['accuracy'])
checkpoint_augmented = ModelCheckpoint('best_model_augmented.h5', save_best_only=True, monitor='val_loss', mode='min', verbose=1)
history_augmented = model_augmented.fit(data_generator.flow(x_train, y_train, batch_size=32), epochs=50, validation_data=(x_val, y_val), callbacks=[checkpoint_augmented])
model_augmented.load_weights('best_model_augmented.h5')
train_loss_augmented, train_accuracy_augmented = model_augmented.evaluate(x_train, y_train)
val_loss_augmented, val_accuracy_augmented = model_augmented.evaluate(x_val, y_val)
```

```python
# Step f: Plot training and validation loss and accuracy for the second model (with data augmentation)
plt.figure(figsize=(12, 4))
plt.subplot(1, 2, 1)
plt.plot(history_augmented.history['loss'], label='Training Loss')
plt.plot(history_augmented.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Model with Data Augmentation')
plt.legend()
plt.grid(True)

plt.subplot(1, 2, 2)
plt.plot(history_augmented.history['accuracy'], label='Training Accuracy')
plt.plot(history_augmented.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy (%)')
plt.title('Model with Data Augmentation')
plt.legend()
plt.grid(True)

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

```python
# Step f: Print training and validation accuracy and loss for the second model (with data augmentation)
print("\nModel with Data Augmentation:")
print("Training Loss: {:.4f}".format(history_augmented.history['loss'][-1]))
print("Training Accuracy: {:.4f}%".format(history_augmented.history['accuracy'][-1] * 100))
print("Validation Loss: {:.4f}".format(history_augmented.history['val_loss'][-1]))
print("Validation Accuracy: {:.4f}%".format(history_augmented.history['val_accuracy'][-1] * 100))

# Step f: Print summary of the second model (with Data Augmentation)
print("\nModel Summary (with Data Augmentation):")
model_augmented.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet152v2_weights_tf_dim_ordering_tf_kernels_notop.h5
234545216/234545216 [==============================] - 7s 0us/step
WARNING:absl:`lr` is deprecated in Keras optimizer, please use `learning_rate` or use the legacy optimizer, e.g.,tf.keras.optimizers.legacy.Adam.
Epoch 1/50
1250/1250 [==============================] - ETA: 0s - loss: 2.1903 - accuracy: 0.1489
Epoch 1: val_loss improved from inf to 2.17104, saving model to best_model_augmented.h5
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3079: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')`.
  saving_api.save_model(
1250/1250 [==============================] - 275s 146ms/step - loss: 2.1903 - accuracy: 0.1489 - val_loss: 2.1710 - val_accuracy: 0.1583
Epoch 2/50
1250/1250 [==============================] - ETA: 0s - loss: 1.9628 - accuracy: 0.2121
Epoch 2: val_loss improved from 2.17104 to 1.87865, saving model to best_model_augmented.h5
1250/1250 [==============================] - 179s 143ms/step - loss: 1.9628 - accuracy: 0.2121 - val_loss: 1.8786 - val_accuracy: 0.2542
Epoch 3/50
1250/1250 [==============================] - ETA: 0s - loss: 1.8580 - accuracy: 0.2725
Epoch 3: val_loss improved from 1.87865 to 1.79871, saving model to best_model_augmented.h5
1250/1250 [==============================] - 185s 148ms/step - loss: 1.8580 - accuracy: 0.2725 - val_loss: 1.7987 - val_accuracy: 0.2711
Epoch 4/50
1250/1250 [==============================] - ETA: 0s - loss: 1.7364 - accuracy: 0.3179
Epoch 4: val_loss did not improve from 1.79871
1250/1250 [==============================] - 174s 139ms/step - loss: 1.7364 - accuracy: 0.3179 - val_loss: 2.1845 - val_accuracy: 0.2643
Epoch 5/50
1250/1250 [==============================] - ETA: 0s - loss: 1.6791 - accuracy: 0.3440
Epoch 5: val_loss did not improve from 1.79871
1250/1250 [==============================] - 171s 137ms/step - loss: 1.6791 - accuracy: 0.3440 - val_loss: 1.9083 - val_accuracy: 0.3516
Epoch 6/50
1250/1250 [==============================] - ETA: 0s - loss: 1.5426 - accuracy: 0.4062
```

```
Epoch 6: val_loss improved from 1.79871 to 1.48656, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 183s 147ms/step - loss: 1.5426 -
accuracy: 0.4062 - val_loss: 1.4866 - val_accuracy: 0.4409
Epoch 7/50
1250/1250 [==============================] - ETA: 0s - loss: 1.4258 -
accuracy: 0.4710
Epoch 7: val_loss improved from 1.48656 to 1.26602, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 188s 150ms/step - loss: 1.4258 -
accuracy: 0.4710 - val_loss: 1.2660 - val_accuracy: 0.5384
Epoch 8/50
1250/1250 [==============================] - ETA: 0s - loss: 1.3222 -
accuracy: 0.5253
Epoch 8: val_loss did not improve from 1.26602
1250/1250 [==============================] - 176s 141ms/step - loss: 1.3222 -
accuracy: 0.5253 - val_loss: 1.7111 - val_accuracy: 0.4583
Epoch 9/50
1250/1250 [==============================] - ETA: 0s - loss: 1.2560 -
accuracy: 0.5548
Epoch 9: val_loss did not improve from 1.26602
1250/1250 [==============================] - 176s 141ms/step - loss: 1.2560 -
accuracy: 0.5548 - val_loss: 1.3473 - val_accuracy: 0.5447
Epoch 10/50
1250/1250 [==============================] - ETA: 0s - loss: 1.1941 -
accuracy: 0.5836
Epoch 10: val_loss did not improve from 1.26602
1250/1250 [==============================] - 177s 142ms/step - loss: 1.1941 -
accuracy: 0.5836 - val_loss: 1.4566 - val_accuracy: 0.5001
Epoch 11/50
1250/1250 [==============================] - ETA: 0s - loss: 1.1316 -
accuracy: 0.6068
Epoch 11: val_loss improved from 1.26602 to 1.05279, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 191s 153ms/step - loss: 1.1316 -
accuracy: 0.6068 - val_loss: 1.0528 - val_accuracy: 0.6223
Epoch 12/50
1250/1250 [==============================] - ETA: 0s - loss: 1.0900 -
accuracy: 0.6267
Epoch 12: val_loss did not improve from 1.05279
1250/1250 [==============================] - 176s 141ms/step - loss: 1.0900 -
accuracy: 0.6267 - val_loss: 1.1745 - val_accuracy: 0.5988
Epoch 13/50
1250/1250 [==============================] - ETA: 0s - loss: 1.0437 -
accuracy: 0.6434
Epoch 13: val_loss improved from 1.05279 to 0.96482, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 193s 154ms/step - loss: 1.0437 -
accuracy: 0.6434 - val_loss: 0.9648 - val_accuracy: 0.6687
Epoch 14/50
1250/1250 [==============================] - ETA: 0s - loss: 1.0189 -
accuracy: 0.6503
Epoch 14: val_loss did not improve from 0.96482
1250/1250 [==============================] - 175s 140ms/step - loss: 1.0189 -
accuracy: 0.6503 - val_loss: 1.0367 - val_accuracy: 0.6449
Epoch 15/50
```

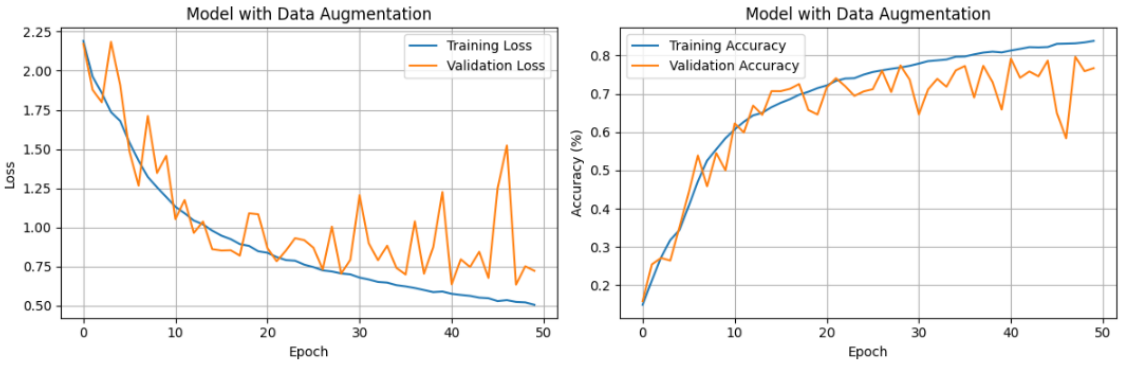
```
1250/1250 [==============================] - ETA: 0s - loss: 0.9789 -
accuracy: 0.6647
Epoch 15: val_loss improved from 0.96482 to 0.86103, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 191s 153ms/step - loss: 0.9789 -
accuracy: 0.6647 - val_loss: 0.8610 - val_accuracy: 0.7066
Epoch 16/50
1250/1250 [==============================] - ETA: 0s - loss: 0.9473 -
accuracy: 0.6762
Epoch 16: val_loss improved from 0.86103 to 0.85211, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 189s 151ms/step - loss: 0.9473 -
accuracy: 0.6762 - val_loss: 0.8521 - val_accuracy: 0.7065
Epoch 17/50
1250/1250 [==============================] - ETA: 0s - loss: 0.9253 -
accuracy: 0.6855
Epoch 17: val_loss did not improve from 0.85211
1250/1250 [==============================] - 174s 139ms/step - loss: 0.9253 -
accuracy: 0.6855 - val_loss: 0.8540 - val_accuracy: 0.7126
Epoch 18/50
1250/1250 [==============================] - ETA: 0s - loss: 0.8930 -
accuracy: 0.6974
Epoch 18: val_loss improved from 0.85211 to 0.81993, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 189s 151ms/step - loss: 0.8930 -
accuracy: 0.6974 - val_loss: 0.8199 - val_accuracy: 0.7252
Epoch 19/50
1250/1250 [==============================] - ETA: 0s - loss: 0.8820 -
accuracy: 0.7049
Epoch 19: val_loss did not improve from 0.81993
1250/1250 [==============================] - 176s 141ms/step - loss: 0.8820 -
accuracy: 0.7049 - val_loss: 1.0898 - val_accuracy: 0.6573
Epoch 20/50
1250/1250 [==============================] - ETA: 0s - loss: 0.8472 -
accuracy: 0.7145
Epoch 20: val_loss did not improve from 0.81993
1250/1250 [==============================] - 174s 139ms/step - loss: 0.8472 -
accuracy: 0.7145 - val_loss: 1.0834 - val_accuracy: 0.6457
Epoch 21/50
1250/1250 [==============================] - ETA: 0s - loss: 0.8377 -
accuracy: 0.7216
Epoch 21: val_loss did not improve from 0.81993
1250/1250 [==============================] - 177s 141ms/step - loss: 0.8377 -
accuracy: 0.7216 - val_loss: 0.8673 - val_accuracy: 0.7164
Epoch 22/50
1250/1250 [==============================] - ETA: 0s - loss: 0.8109 -
accuracy: 0.7333
Epoch 22: val_loss improved from 0.81993 to 0.78372, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 189s 151ms/step - loss: 0.8109 -
accuracy: 0.7333 - val_loss: 0.7837 - val_accuracy: 0.7403
Epoch 23/50
1250/1250 [==============================] - ETA: 0s - loss: 0.7911 -
accuracy: 0.7395
Epoch 23: val_loss did not improve from 0.78372
1250/1250 [==============================] - 173s 139ms/step - loss: 0.7911 -
accuracy: 0.7395 - val_loss: 0.8516 - val_accuracy: 0.7195
```

```
Epoch 24/50
1250/1250 [==============================] - ETA: 0s - loss: 0.7868 -
accuracy: 0.7405
Epoch 24: val_loss did not improve from 0.78372
1250/1250 [==============================] - 174s 139ms/step - loss: 0.7868 -
accuracy: 0.7405 - val_loss: 0.9313 - val_accuracy: 0.6943
Epoch 25/50
1250/1250 [==============================] - ETA: 0s - loss: 0.7619 -
accuracy: 0.7498
Epoch 25: val_loss did not improve from 0.78372
1250/1250 [==============================] - 173s 138ms/step - loss: 0.7619 -
accuracy: 0.7498 - val_loss: 0.9183 - val_accuracy: 0.7058
Epoch 26/50
1250/1250 [==============================] - ETA: 0s - loss: 0.7463 -
accuracy: 0.7565
Epoch 26: val_loss did not improve from 0.78372
1250/1250 [==============================] - 174s 139ms/step - loss: 0.7463 -
accuracy: 0.7565 - val_loss: 0.8697 - val_accuracy: 0.7116
Epoch 27/50
1250/1250 [==============================] - ETA: 0s - loss: 0.7253 -
accuracy: 0.7608
Epoch 27: val_loss improved from 0.78372 to 0.73048, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 189s 151ms/step - loss: 0.7253 -
accuracy: 0.7608 - val_loss: 0.7305 - val_accuracy: 0.7601
Epoch 28/50
1250/1250 [==============================] - ETA: 0s - loss: 0.7182 -
accuracy: 0.7648
Epoch 28: val_loss did not improve from 0.73048
1250/1250 [==============================] - 174s 139ms/step - loss: 0.7182 -
accuracy: 0.7648 - val_loss: 1.0052 - val_accuracy: 0.7043
Epoch 29/50
1250/1250 [==============================] - ETA: 0s - loss: 0.7059 -
accuracy: 0.7683
Epoch 29: val_loss improved from 0.73048 to 0.70456, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 189s 151ms/step - loss: 0.7059 -
accuracy: 0.7683 - val_loss: 0.7046 - val_accuracy: 0.7739
Epoch 30/50
1250/1250 [==============================] - ETA: 0s - loss: 0.7003 -
accuracy: 0.7728
Epoch 30: val_loss did not improve from 0.70456
1250/1250 [==============================] - 173s 139ms/step - loss: 0.7003 -
accuracy: 0.7728 - val_loss: 0.7929 - val_accuracy: 0.7368
Epoch 31/50
1250/1250 [==============================] - ETA: 0s - loss: 0.6795 -
accuracy: 0.7785
Epoch 31: val_loss did not improve from 0.70456
1250/1250 [==============================] - 174s 139ms/step - loss: 0.6795 -
accuracy: 0.7785 - val_loss: 1.2060 - val_accuracy: 0.6457
Epoch 32/50
1250/1250 [==============================] - ETA: 0s - loss: 0.6672 -
accuracy: 0.7848
Epoch 32: val_loss did not improve from 0.70456
1250/1250 [==============================] - 171s 137ms/step - loss: 0.6672 -
accuracy: 0.7848 - val_loss: 0.8987 - val_accuracy: 0.7109
Epoch 33/50
```

```
1250/1250 [==============================] - ETA: 0s - loss: 0.6516 -
accuracy: 0.7869
Epoch 33: val_loss did not improve from 0.70456
1250/1250 [==============================] - 173s 139ms/step - loss: 0.6516 -
accuracy: 0.7869 - val_loss: 0.7901 - val_accuracy: 0.7387
Epoch 34/50
1250/1250 [==============================] - ETA: 0s - loss: 0.6473 -
accuracy: 0.7892
Epoch 34: val_loss did not improve from 0.70456
1250/1250 [==============================] - 171s 137ms/step - loss: 0.6473 -
accuracy: 0.7892 - val_loss: 0.8832 - val_accuracy: 0.7181
Epoch 35/50
1250/1250 [==============================] - ETA: 0s - loss: 0.6313 -
accuracy: 0.7965
Epoch 35: val_loss did not improve from 0.70456
1250/1250 [==============================] - 174s 139ms/step - loss: 0.6313 -
accuracy: 0.7965 - val_loss: 0.7414 - val_accuracy: 0.7606
Epoch 36/50
1250/1250 [==============================] - ETA: 0s - loss: 0.6233 -
accuracy: 0.7972
Epoch 36: val_loss improved from 0.70456 to 0.69908, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 192s 153ms/step - loss: 0.6233 -
accuracy: 0.7972 - val_loss: 0.6991 - val_accuracy: 0.7723
Epoch 37/50
1250/1250 [==============================] - ETA: 0s - loss: 0.6128 -
accuracy: 0.8026
Epoch 37: val_loss did not improve from 0.69908
1250/1250 [==============================] - 173s 139ms/step - loss: 0.6128 -
accuracy: 0.8026 - val_loss: 1.0388 - val_accuracy: 0.6900
Epoch 38/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5998 -
accuracy: 0.8072
Epoch 38: val_loss did not improve from 0.69908
1250/1250 [==============================] - 174s 139ms/step - loss: 0.5998 -
accuracy: 0.8072 - val_loss: 0.7043 - val_accuracy: 0.7727
Epoch 39/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5868 -
accuracy: 0.8096
Epoch 39: val_loss did not improve from 0.69908
1250/1250 [==============================] - 177s 142ms/step - loss: 0.5868 -
accuracy: 0.8096 - val_loss: 0.8743 - val_accuracy: 0.7303
Epoch 40/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5904 -
accuracy: 0.8077
Epoch 40: val_loss did not improve from 0.69908
1250/1250 [==============================] - 177s 142ms/step - loss: 0.5904 -
accuracy: 0.8077 - val_loss: 1.2257 - val_accuracy: 0.6585
Epoch 41/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5761 -
accuracy: 0.8126
Epoch 41: val_loss improved from 0.69908 to 0.63643, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 188s 150ms/step - loss: 0.5761 -
accuracy: 0.8126 - val_loss: 0.6364 - val_accuracy: 0.7912
Epoch 42/50
```

```
1250/1250 [==============================] - ETA: 0s - loss: 0.5689 -
accuracy: 0.8169
Epoch 42: val_loss did not improve from 0.63643
1250/1250 [==============================] - 178s 142ms/step - loss: 0.5689 -
accuracy: 0.8169 - val_loss: 0.7956 - val_accuracy: 0.7417
Epoch 43/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5629 -
accuracy: 0.8215
Epoch 43: val_loss did not improve from 0.63643
1250/1250 [==============================] - 174s 139ms/step - loss: 0.5629 -
accuracy: 0.8215 - val_loss: 0.7473 - val_accuracy: 0.7579
Epoch 44/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5514 -
accuracy: 0.8208
Epoch 44: val_loss did not improve from 0.63643
1250/1250 [==============================] - 172s 138ms/step - loss: 0.5514 -
accuracy: 0.8208 - val_loss: 0.8438 - val_accuracy: 0.7454
Epoch 45/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5476 -
accuracy: 0.8219
Epoch 45: val_loss did not improve from 0.63643
1250/1250 [==============================] - 171s 136ms/step - loss: 0.5476 -
accuracy: 0.8219 - val_loss: 0.6774 - val_accuracy: 0.7862
Epoch 46/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5293 -
accuracy: 0.8300
Epoch 46: val_loss did not improve from 0.63643
1250/1250 [==============================] - 173s 138ms/step - loss: 0.5293 -
accuracy: 0.8300 - val_loss: 1.2502 - val_accuracy: 0.6492
Epoch 47/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5354 -
accuracy: 0.8306
Epoch 47: val_loss did not improve from 0.63643
1250/1250 [==============================] - 171s 137ms/step - loss: 0.5354 -
accuracy: 0.8306 - val_loss: 1.5224 - val_accuracy: 0.5833
Epoch 48/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5241 -
accuracy: 0.8313
Epoch 48: val_loss improved from 0.63643 to 0.63485, saving model to
best_model_augmented.h5
1250/1250 [==============================] - 184s 147ms/step - loss: 0.5241 -
accuracy: 0.8313 - val_loss: 0.6349 - val_accuracy: 0.7962
Epoch 49/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5212 -
accuracy: 0.8338
Epoch 49: val_loss did not improve from 0.63485
1250/1250 [==============================] - 170s 136ms/step - loss: 0.5212 -
accuracy: 0.8338 - val_loss: 0.7516 - val_accuracy: 0.7586
Epoch 50/50
1250/1250 [==============================] - ETA: 0s - loss: 0.5060 -
accuracy: 0.8378
Epoch 50: val_loss did not improve from 0.63485
1250/1250 [==============================] - 174s 139ms/step - loss: 0.5060 -
accuracy: 0.8378 - val_loss: 0.7224 - val_accuracy: 0.7666
1250/1250 [==============================] - 39s 31ms/step - loss: 0.4508 -
accuracy: 0.8487
```

```
313/313 [==============================] - 10s 33ms/step - loss: 0.6349 -
accuracy: 0.7962
```




```
Model with Data Augmentation:
Training Loss: 0.5060
Training Accuracy: 83.7750%
Validation Loss: 0.7224
Validation Accuracy: 76.6600%

Model Summary (with Data Augmentation):
Model: "sequential_1"
```

Layer (type)	Output Shape	Param #
resnet152v2 (Functional)	(None, 1, 1, 2048)	58331648
conv2d (Conv2D)	(None, 1, 1, 32)	589856
activation (Activation)	(None, 1, 1, 32)	0
conv2d_1 (Conv2D)	(None, 1, 1, 32)	9248
activation_1 (Activation)	(None, 1, 1, 32)	0
max_pooling2d_3 (MaxPooling2D)	(None, 1, 1, 32)	0
conv2d_2 (Conv2D)	(None, 1, 1, 64)	18496
activation_2 (Activation)	(None, 1, 1, 64)	0
conv2d_3 (Conv2D)	(None, 1, 1, 64)	36928
activation_3 (Activation)	(None, 1, 1, 64)	0
max_pooling2d_4 (MaxPooling2D)	(None, 1, 1, 64)	0
flatten (Flatten)	(None, 64)	0
dense (Dense)	(None, 512)	33280
dropout (Dropout)	(None, 512)	0

```
 dense_1 (Dense)             (None, 10)                5130

=================================================================
Total params: 59024586 (225.16 MB)
Trainable params: 58880842 (224.61 MB)
Non-trainable params: 143744 (561.50 KB)
_________________________________________________________________
```

```
[ ] # Evaluate the model with test data
    test_loss, test_accuracy = model_augmented.evaluate(x_test, y_test)
    print(f'Test Loss: {test_loss:.4f}, Test Accuracy: {test_accuracy*100:.2f}%')

    313/313 [==============================] - 10s 32ms/step - loss: 0.6532 - accuracy: 0.7895
    Test Loss: 0.6532, Test Accuracy: 78.95%
```