Harishwar Reddy Abbareddy

ICP-7

YouTube Link: https://youtu.be/-r-dAV9q600
GitHub Link: https://github.com/Harishwar-reddi/ICP-7

```
# Mount  Google Drive
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
lu'))
```

```
import numpy as np
from tensorflow.keras.datasets import cifar10
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPooling2D
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import LearningRateScheduler
import matplotlib.pyplot as plt

# Learning rate schedule function
def lr_schedule(epoch):
    initial_lr = 0.01
    decay = initial_lr / epochs
    lrate = initial_lr * (1 / (1 + decay * epoch))
    return lrate

# Fix random seed for reproducibility
np.random.seed(7)

# Load data
(X_train, y_train), (X_test, y_test) = cifar10.load_data()

# Normalize inputs from 0-255 to 0.0-1.0
X_train = X_train.astype('float32') / 255.0
X_test = X_test.astype('float32') / 255.0

# One-hot encode outputs
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)
num_classes = y_test.shape[1]
```

```
# Create the modified model
model = Sequential()

# Add layers according to the new architecture
model.add(Conv2D(32, (3, 3), input_shape=(32, 32, 3), padding='same', activation='relu'))
model.add(Dropout(0.2))
model.add(Conv2D(32, (3, 3), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
model.add(Dropout(0.2))
model.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu', padding='same'))
model.add(Dropout(0.2))
model.add(Conv2D(128, (3, 3), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dropout(0.2))
model.add(Dense(1024, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))

# Compile the model
epochs = 25
sgd = SGD(learning_rate=0.01, momentum=0.9, nesterov=False)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
print(model.summary())

# Create LearningRateScheduler callback
callbacks = [LearningRateScheduler(lr_schedule)]
```

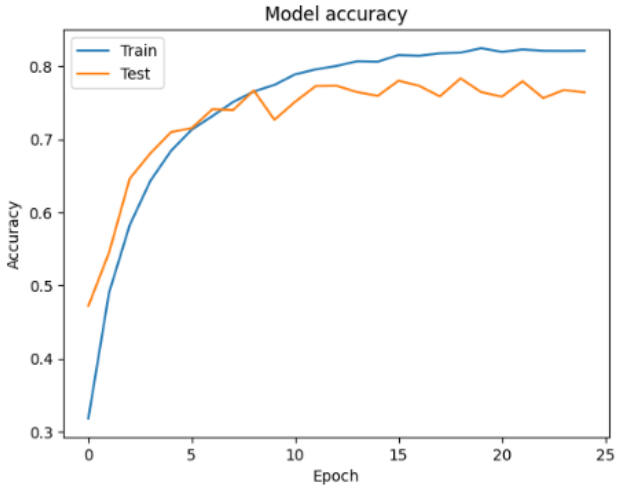
```
# Train the model
history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=epochs, batch_size=32, callbacks=callbacl

# Evaluate the model
scores = model.evaluate(X_test, y_test, verbose=0)
print("Accuracy: %.2f%%" % (scores[1] * 100))

# Predict the first 4 images in the test set
predictions = model.predict(X_test[:4])
predicted_labels = np.argmax(predictions, axis=1)
actual_labels = np.argmax(y_test[:4], axis=1)

# Compare predicted and actual labels
print("Predicted labels: ", predicted_labels)
print("Actual labels: ", actual_labels)

# Plot training & validation accuracy values
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()

# Plot training & validation loss values
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```

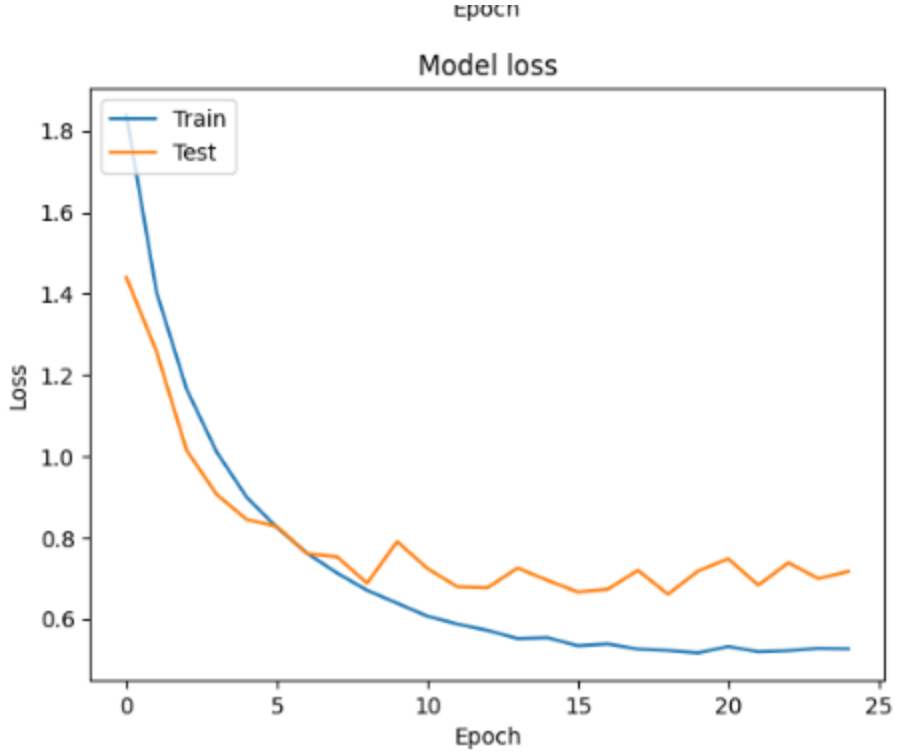
```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 [==============================] - 2s 0us/step
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 32, 32, 32)        896

 dropout (Dropout)           (None, 32, 32, 32)        0

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

 max_pooling2d (MaxPooling2  (None, 16, 16, 32)        0
 D)

 conv2d_2 (Conv2D)           (None, 16, 16, 64)        18496

 dropout_1 (Dropout)         (None, 16, 16, 64)        0

 conv2d_3 (Conv2D)           (None, 16, 16, 64)        36928

 max_pooling2d_1 (MaxPoolin  (None, 8, 8, 64)          0
 g2D)

 conv2d_4 (Conv2D)           (None, 8, 8, 128)         73856

 dropout_2 (Dropout)         (None, 8, 8, 128)         0

 conv2d_5 (Conv2D)           (None, 8, 8, 128)         147584

 max_pooling2d_2 (MaxPoolin  (None, 4, 4, 128)         0
 g2D)

 flatten (Flatten)           (None, 2048)              0

 dropout_3 (Dropout)         (None, 2048)              0

 dense (Dense)               (None, 1024)              2098176

 dropout_4 (Dropout)         (None, 1024)              0

 dense_1 (Dense)             (None, 512)               524800

 dropout_5 (Dropout)         (None, 512)               0
```

```
 dropout_4 (Dropout)         (None, 1024)              0

 dense_1 (Dense)             (None, 512)               524800

 dropout_5 (Dropout)         (None, 512)               0

 dense_2 (Dense)             (None, 10)                5130

=================================================================
Total params: 2915114 (11.12 MB)
Trainable params: 2915114 (11.12 MB)
Non-trainable params: 0 (0.00 Byte)
```

```
_________________________________________________________________
None
Epoch 1/25
1563/1563 [==============================] - 29s 9ms/step - loss: 1.8387 - accuracy: 0.3181 - val_loss: 1.4396 - val_accuracy: 0.4721 - lr: 0.0100
Epoch 2/25
1563/1563 [==============================] - 16s 10ms/step - loss: 1.4039 - accuracy: 0.4901 - val_loss: 1.2575 - val_accuracy: 0.5451 - lr: 0.010
Epoch 3/25
1563/1563 [==============================] - 14s 9ms/step - loss: 1.1650 - accuracy: 0.5828 - val_loss: 1.0145 - val_accuracy: 0.6464 - lr: 0.0100
Epoch 4/25
1563/1563 [==============================] - 14s 9ms/step - loss: 1.0102 - accuracy: 0.6434 - val_loss: 0.9057 - val_accuracy: 0.6811 - lr: 0.0100
Epoch 5/25
1563/1563 [==============================] - 14s 9ms/step - loss: 0.8988 - accuracy: 0.6845 - val_loss: 0.8442 - val_accuracy: 0.7101 - lr: 0.0100
Epoch 6/25
1563/1563 [==============================] - 13s 8ms/step - loss: 0.8248 - accuracy: 0.7135 - val_loss: 0.8273 - val_accuracy: 0.7156 - lr: 0.0100
Epoch 7/25
1563/1563 [==============================] - 13s 9ms/step - loss: 0.7615 - accuracy: 0.7321 - val_loss: 0.7603 - val_accuracy: 0.7415 - lr: 0.0100
Epoch 8/25
1563/1563 [==============================] - 13s 8ms/step - loss: 0.7125 - accuracy: 0.7514 - val_loss: 0.7524 - val_accuracy: 0.7403 - lr: 0.0100
Epoch 9/25
1563/1563 [==============================] - 13s 8ms/step - loss: 0.6700 - accuracy: 0.7657 - val_loss: 0.6875 - val_accuracy: 0.7671 - lr: 0.0100
Epoch 10/25
1563/1563 [==============================] - 14s 9ms/step - loss: 0.6382 - accuracy: 0.7748 - val_loss: 0.7900 - val_accuracy: 0.7270 - lr: 0.0100
Epoch 11/25
1563/1563 [==============================] - 14s 9ms/step - loss: 0.6066 - accuracy: 0.7892 - val_loss: 0.7240 - val_accuracy: 0.7517 - lr: 0.0100
Epoch 12/25
1563/1563 [==============================] - 13s 9ms/step - loss: 0.5870 - accuracy: 0.7960 - val_loss: 0.6787 - val_accuracy: 0.7732 - lr: 0.0100
Epoch 13/25
1563/1563 [==============================] - 13s 8ms/step - loss: 0.5714 - accuracy: 0.8007 - val_loss: 0.6763 - val_accuracy: 0.7736 - lr: 0.0100
Epoch 14/25
1563/1563 [==============================] - 13s 8ms/step - loss: 0.5513 - accuracy: 0.8071 - val_loss: 0.7250 - val_accuracy: 0.7649 - lr: 0.0099
Epoch 15/25
1563/1563 [==============================] - 13s 9ms/step - loss: 0.5534 - accuracy: 0.8066 - val_loss: 0.6945 - val_accuracy: 0.7598 - lr: 0.0099
Epoch 16/25
1563/1563 [==============================] - 13s 8ms/step - loss: 0.5336 - accuracy: 0.8156 - val_loss: 0.6660 - val_accuracy: 0.7805 - lr: 0.0099
Epoch 17/25
1563/1563 [==============================] - 14s 9ms/step - loss: 0.5381 - accuracy: 0.8147 - val_loss: 0.6727 - val_accuracy: 0.7735 - lr: 0.0099
Epoch 18/25
1563/1563 [==============================] - 13s 9ms/step - loss: 0.5253 - accuracy: 0.8180 - val_loss: 0.7192 - val_accuracy: 0.7590 - lr: 0.0099
Epoch 19/25
1563/1563 [==============================] - 13s 9ms/step - loss: 0.5220 - accuracy: 0.8189 - val_loss: 0.6604 - val_accuracy: 0.7837 - lr: 0.0099
Epoch 20/25
1563/1563 [==============================] - 14s 9ms/step - loss: 0.5161 - accuracy: 0.8250 - val_loss: 0.7175 - val_accuracy: 0.7649 - lr: 0.0099
Epoch 21/25
1563/1563 [==============================] - 14s 9ms/step - loss: 0.5317 - accuracy: 0.8200 - val_loss: 0.7477 - val_accuracy: 0.7586 - lr: 0.0099
```

```
Epoch 21/25
1563/1563 [==============================] - 14s 9ms/step - loss: 0.5317 - accuracy: 0.8200 - val_loss: 0.7477 - val_accuracy: 0.7586 - lr: 0.0099
Epoch 22/25
1563/1563 [==============================] - 14s 9ms/step - loss: 0.5189 - accuracy: 0.8233 - val_loss: 0.6826 - val_accuracy: 0.7798 - lr: 0.0099
Epoch 23/25
1563/1563 [==============================] - 13s 8ms/step - loss: 0.5215 - accuracy: 0.8215 - val_loss: 0.7378 - val_accuracy: 0.7567 - lr: 0.0099
Epoch 24/25
1563/1563 [==============================] - 13s 8ms/step - loss: 0.5271 - accuracy: 0.8213 - val_loss: 0.6988 - val_accuracy: 0.7678 - lr: 0.0099
Epoch 25/25
1563/1563 [==============================] - 13s 8ms/step - loss: 0.5260 - accuracy: 0.8216 - val_loss: 0.7166 - val_accuracy: 0.7645 - lr: 0.0099
Accuracy: 76.45%
1/1 [==============================] - 0s 343ms/step
Predicted labels:  [3 8 8 0]
Actual labels:  [3 8 8 0]
```




Epoch
Model loss
Train
Test
1.8
1.6
1.4
1.2
1.0
0.8
0.6
Loss
0
5
10
15
20
25
Epoch