Neural Networks and deep learning – ICP4

KARTIK ANUMOLU

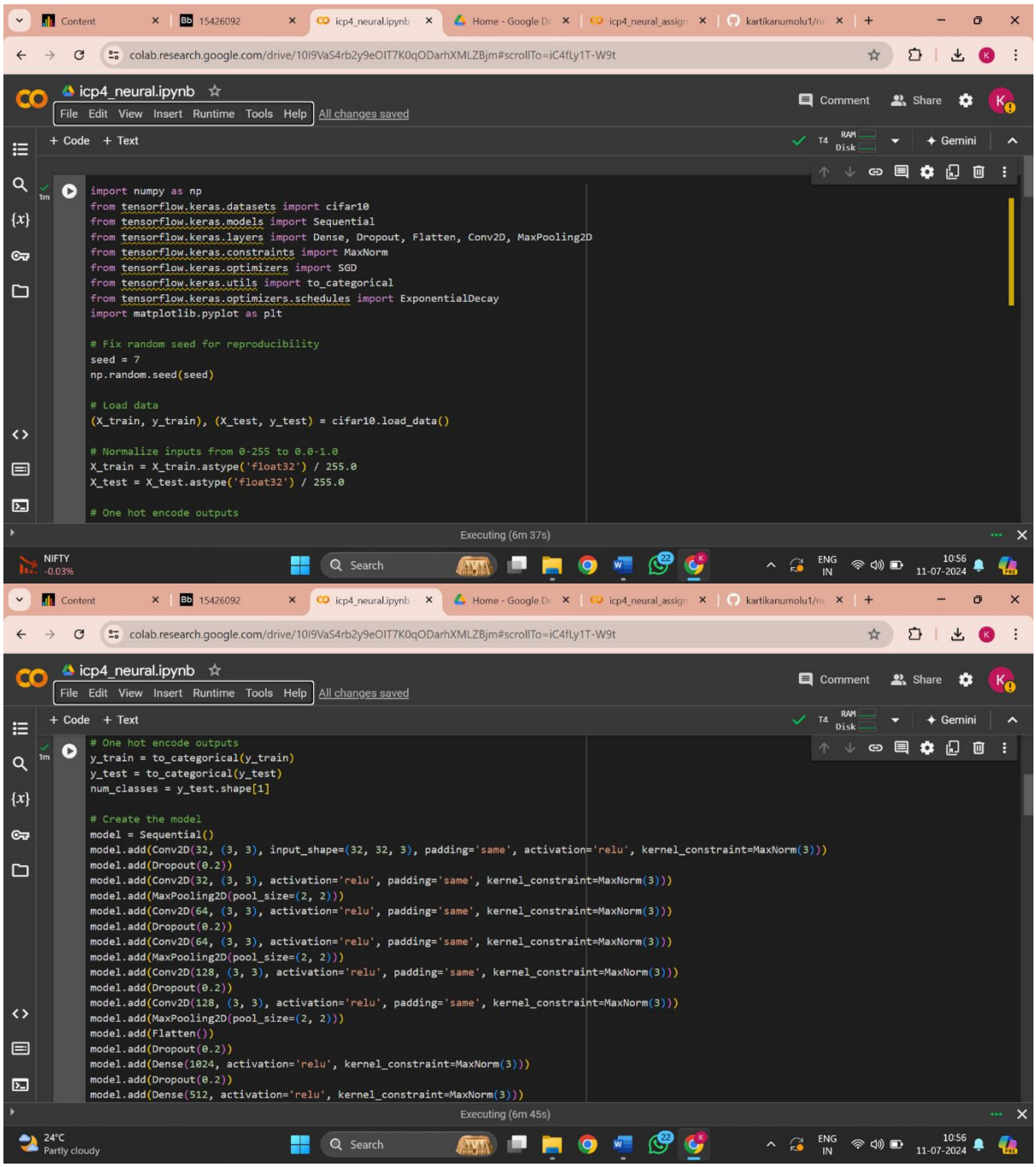
Student ID : 700758573

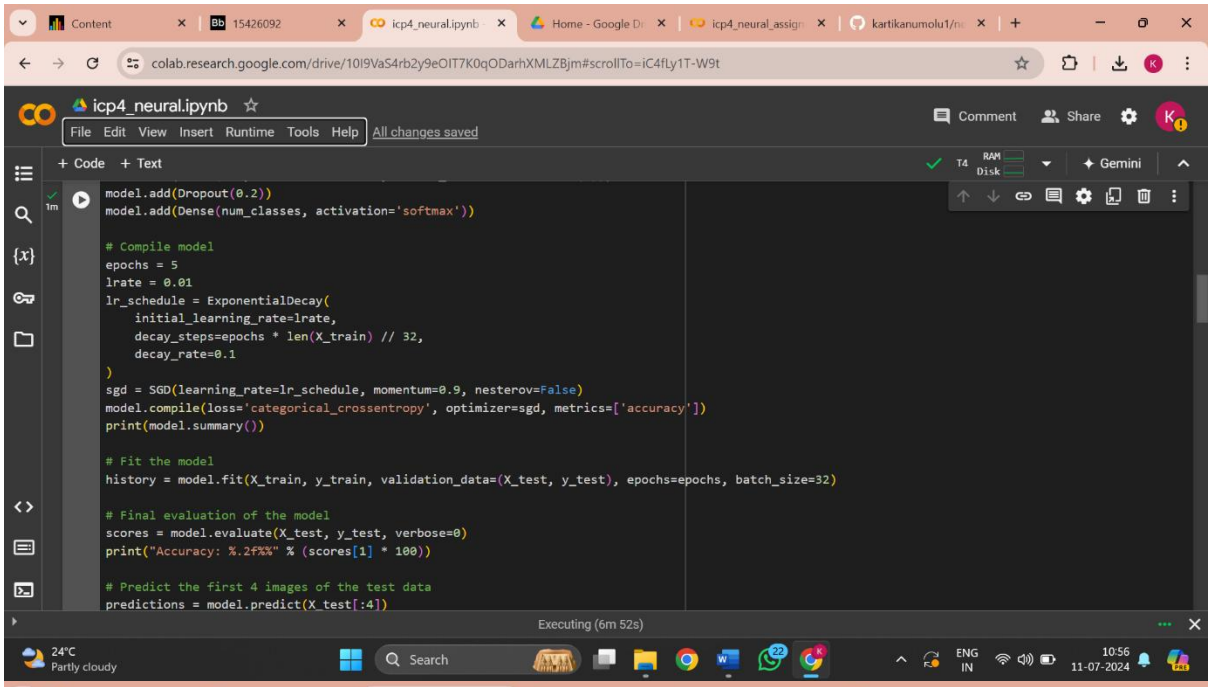
Github link:https://github.com/kartikanumolu1/neural-assignment4.git

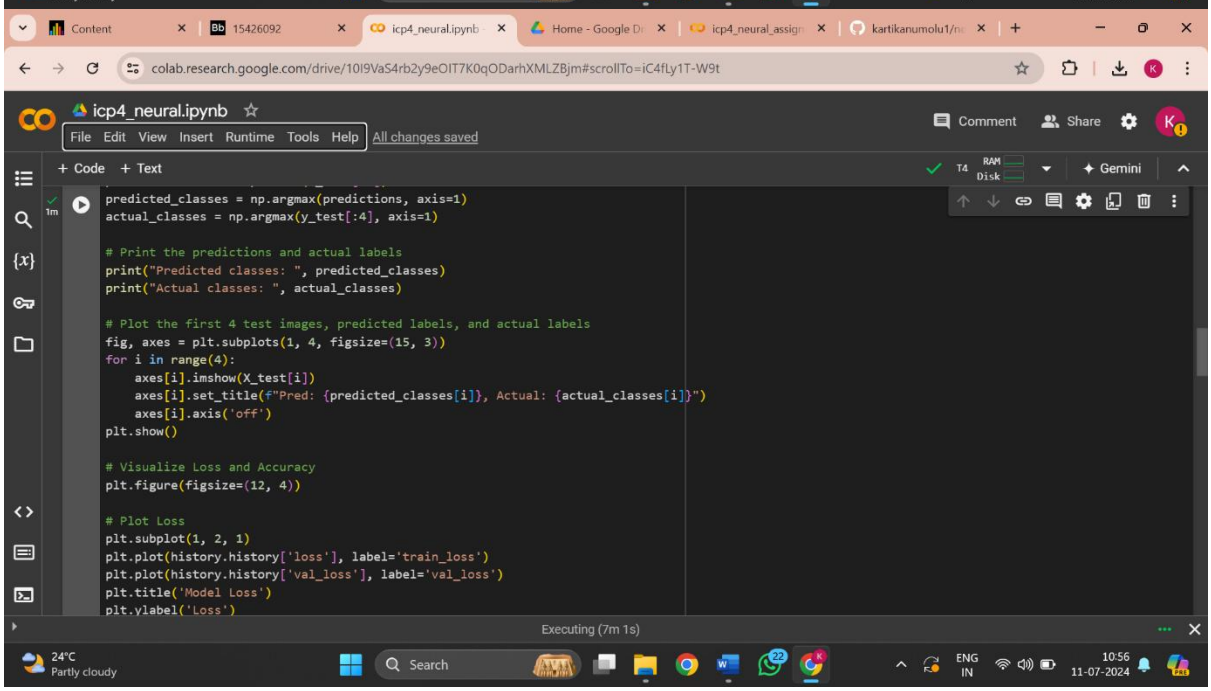
Video link:
https://drive.google.com/file/d/1J4_9327I_rSTf4idXSEUptxCaUVuWUdM/view?usp=drive_link

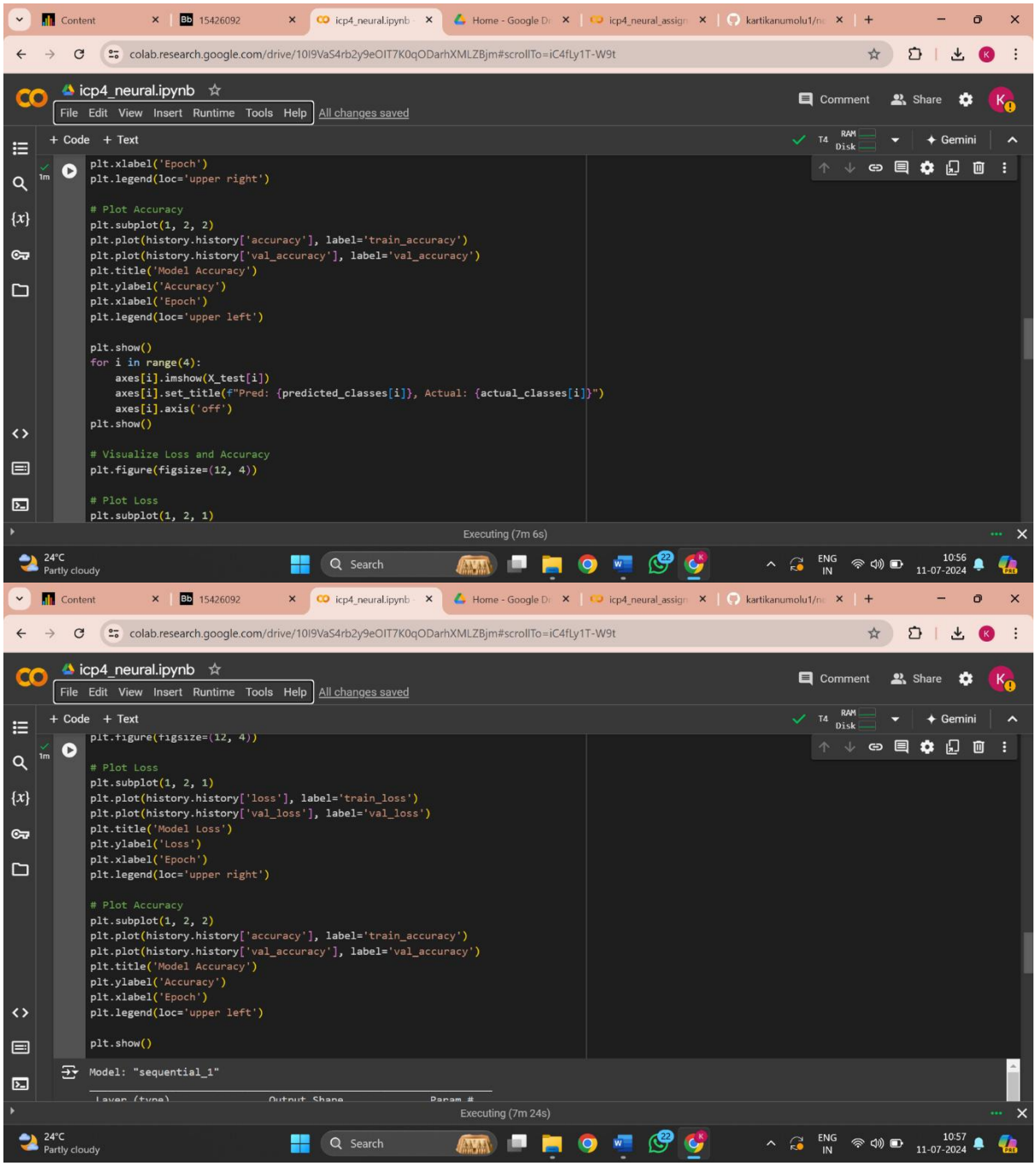
1. Follow the instruction below and then report how the performance changed.(apply all at once)
   • Convolutional input layer, 32 feature maps with a size of 3×3 and a rectifier activation function.
   • Dropout layer at 20%.
   • Convolutional layer, 32 feature maps with a size of 3×3 and a rectifier activation function.
   • Max Pool layer with size 2×2.
   • Convolutional layer, 64 feature maps with a size of 3×3 and a rectifier activation function.
   • Dropout layer at 20%.
   • Convolutional layer, 64 feature maps with a size of 3×3 and a rectifier activation function.
   • Max Pool layer with size 2×2.
   • Convolutional layer, 128 feature maps with a size of 3×3 and a rectifier activation function.
   • Dropout layer at 20%.
   • Convolutional layer,128 feature maps with a size of 3×3 and a rectifier activation function.
   • Max Pool layer with size 2×2.
   • Flatten layer.
   • Dropout layer at 20%.
   • Fully connected layer with 1024 units and a rectifier activation function.
   • Dropout layer at 20%.
   • Fully connected layer with 512 units and a rectifier activation function.
   • Dropout layer at 20%.
   • Fully connected output layer with 10 units and a Softmax activation function Did the performance change?
2. Predict the first 4 images of the test data using the above model. Then, compare with the actual label for those 4 images to check whether or not the model has predicted correctly.
3. Visualize Loss and Accuracy using the history object

Output:

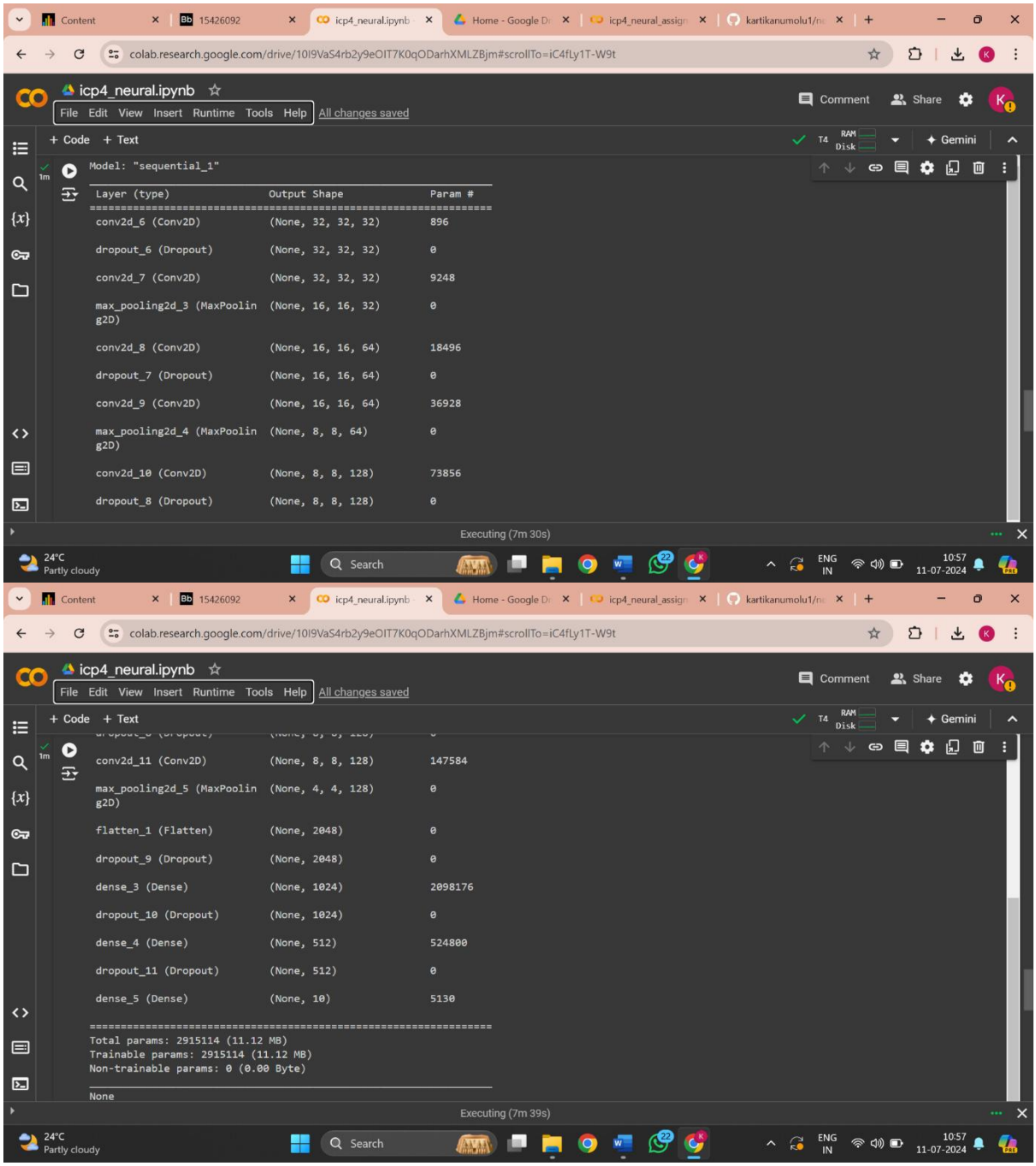

icp4_neural.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Gemini
import numpy as np
from tensorflow.keras.datasets import cifar10
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPooling2D
from tensorflow.keras.constraints import MaxNorm
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.optimizers.schedules import ExponentialDecay
import matplotlib.pyplot as plt
# Fix random seed for reproducibility
seed = 7
np.random.seed(seed)
# 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
Executing (6m 37s)
# One hot encode outputs
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)
num_classes = y_test.shape[1]
# Create the model
model = Sequential()
model.add(Conv2D(32, (3, 3), input_shape=(32, 32, 3), padding='same', activation='relu', kernel_constraint=MaxNorm(3)))
model.add(Dropout(0.2))
model.add(Conv2D(32, (3, 3), activation='relu', padding='same', kernel_constraint=MaxNorm(3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu', padding='same', kernel_constraint=MaxNorm(3)))
model.add(Dropout(0.2))
model.add(Conv2D(64, (3, 3), activation='relu', padding='same', kernel_constraint=MaxNorm(3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu', padding='same', kernel_constraint=MaxNorm(3)))
model.add(Dropout(0.2))
model.add(Conv2D(128, (3, 3), activation='relu', padding='same', kernel_constraint=MaxNorm(3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dropout(0.2))
model.add(Dense(1024, activation='relu', kernel_constraint=MaxNorm(3)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='relu', kernel_constraint=MaxNorm(3)))
Executing (6m 45s)


icp4_neural.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Gemini
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
# Compile model
epochs = 5
lrate = 0.01
lr_schedule = ExponentialDecay(
initial_learning_rate=lrate,
decay_steps=epochs * len(X_train) // 32,
decay_rate=0.1
)
sgd = SGD(learning_rate=lr_schedule, momentum=0.9, nesterov=False)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
print(model.summary())
# Fit the model
history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=epochs, batch_size=32)
# Final evaluation of the model
scores = model.evaluate(X_test, y_test, verbose=0)
print("Accuracy: %.2f%%" % (scores[1] * 100))
# Predict the first 4 images of the test data
predictions = model.predict(X_test[:4])
Executing (6m 52s)

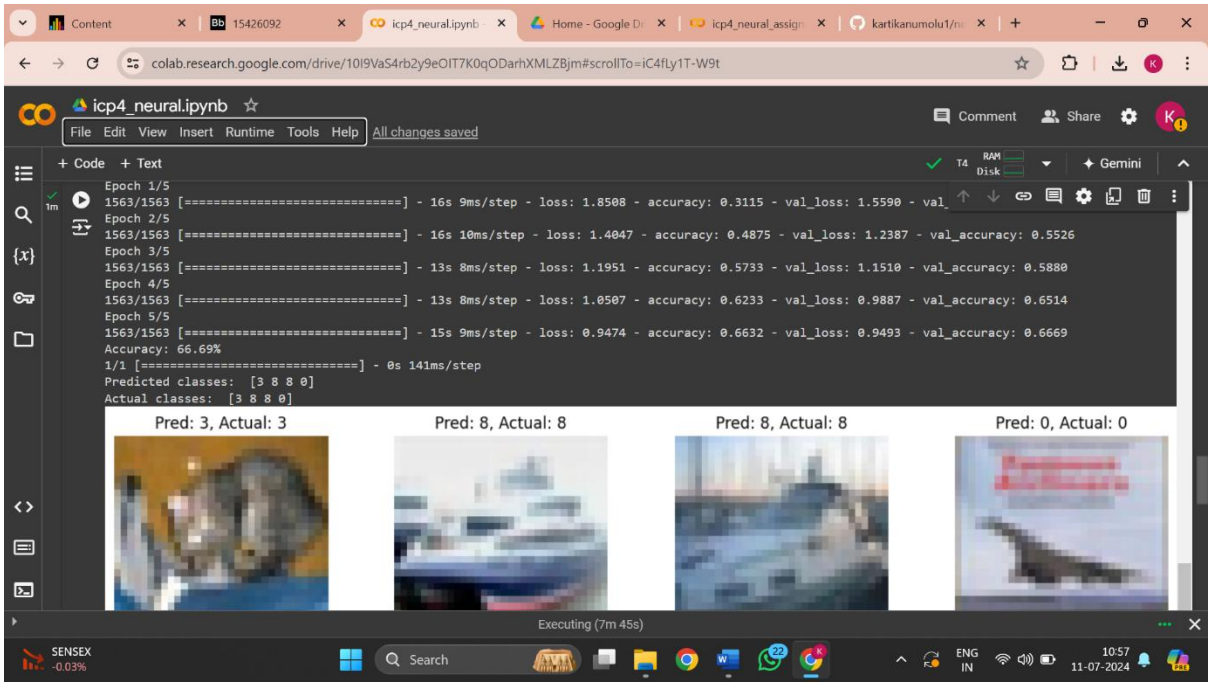

icp4_neural.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Gemini
predicted_classes = np.argmax(predictions, axis=1)
actual_classes = np.argmax(y_test[:4], axis=1)
# Print the predictions and actual labels
print("Predicted classes: ", predicted_classes)
print("Actual classes: ", actual_classes)
# Plot the first 4 test images, predicted labels, and actual labels
fig, axes = plt.subplots(1, 4, figsize=(15, 3))
for i in range(4):
axes[i].imshow(X_test[i])
axes[i].set_title(f"Pred: {predicted_classes[i]}, Actual: {actual_classes[i]}")
axes[i].axis('off')
plt.show()
# Visualize Loss and Accuracy
plt.figure(figsize=(12, 4))
# Plot Loss
plt.subplot(1, 2, 1)
plt.plot(history.history['loss'], label='train_loss')
plt.plot(history.history['val_loss'], label='val_loss')
plt.title('Model Loss')
plt.ylabel('Loss')
Executing (7m 1s)

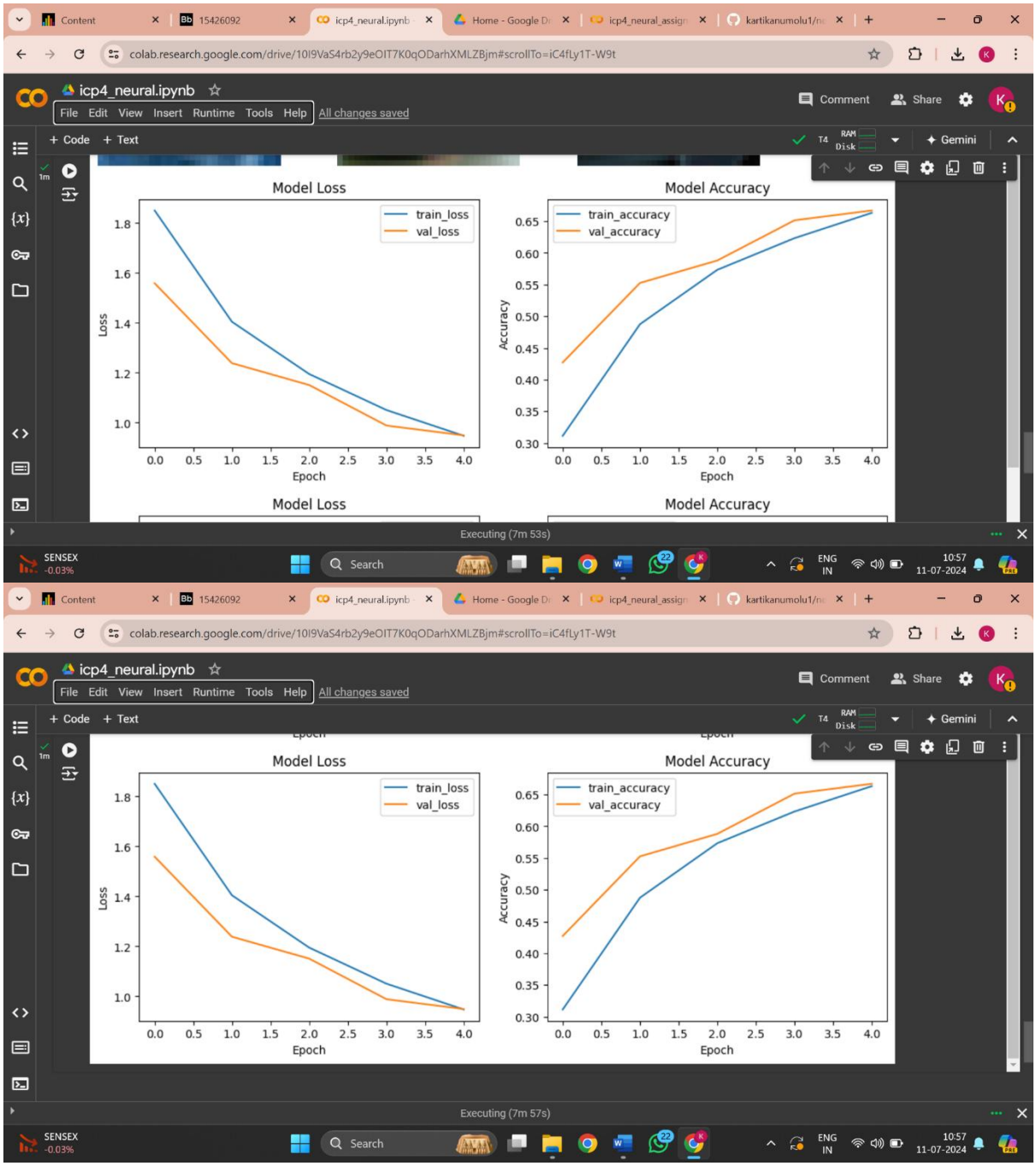

icp4_neural.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Gemini
plt.xlabel('Epoch')
plt.legend(loc='upper right')
# Plot Accuracy
plt.subplot(1, 2, 2)
plt.plot(history.history['accuracy'], label='train_accuracy')
plt.plot(history.history['val_accuracy'], label='val_accuracy')
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(loc='upper left')
plt.show()
for i in range(4):
axes[i].imshow(X_test[i])
axes[i].set_title(f"Pred: {predicted_classes[i]}, Actual: {actual_classes[i]}")
axes[i].axis('off')
plt.show()
# Visualize Loss and Accuracy
plt.figure(figsize=(12, 4))
# Plot Loss
plt.subplot(1, 2, 1)
Executing (7m 6s)
24°C
Partly cloudy
Search
ENG
IN
10:56
11-07-2024

icp4_neural.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Gemini
plt.figure(figsize=(12, 4))
# Plot Loss
plt.subplot(1, 2, 1)
plt.plot(history.history['loss'], label='train_loss')
plt.plot(history.history['val_loss'], label='val_loss')
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(loc='upper right')
# Plot Accuracy
plt.subplot(1, 2, 2)
plt.plot(history.history['accuracy'], label='train_accuracy')
plt.plot(history.history['val_accuracy'], label='val_accuracy')
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(loc='upper left')
plt.show()
Model: "sequential_1"
Executing (7m 24s)
24°C
Partly cloudy
Search
ENG
IN
10:57
11-07-2024


icp4_neural.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Gemini
Model: "sequential_1"
Layer (type) Output Shape Param #
conv2d_6 (Conv2D) (None, 32, 32, 32) 896
dropout_6 (Dropout) (None, 32, 32, 32) 0
conv2d_7 (Conv2D) (None, 32, 32, 32) 9248
max_pooling2d_3 (MaxPoolin g2D) (None, 16, 16, 32) 0
conv2d_8 (Conv2D) (None, 16, 16, 64) 18496
dropout_7 (Dropout) (None, 16, 16, 64) 0
conv2d_9 (Conv2D) (None, 16, 16, 64) 36928
max_pooling2d_4 (MaxPoolin g2D) (None, 8, 8, 64) 0
conv2d_10 (Conv2D) (None, 8, 8, 128) 73856
dropout_8 (Dropout) (None, 8, 8, 128) 0
Executing (7m 30s)
24°C Partly cloudy
Search
ENG IN
10:57 11-07-2024

icp4_neural.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Gemini
conv2d_11 (Conv2D) (None, 8, 8, 128) 147584
max_pooling2d_5 (MaxPoolin g2D) (None, 4, 4, 128) 0
flatten_1 (Flatten) (None, 2048) 0
dropout_9 (Dropout) (None, 2048) 0
dense_3 (Dense) (None, 1024) 2098176
dropout_10 (Dropout) (None, 1024) 0
dense_4 (Dense) (None, 512) 524800
dropout_11 (Dropout) (None, 512) 0
dense_5 (Dense) (None, 10) 5130
Total params: 2915114 (11.12 MB)
Trainable params: 2915114 (11.12 MB)
Non-trainable params: 0 (0.00 Byte)
None
Executing (7m 39s)
24°C Partly cloudy
Search
ENG IN
10:57 11-07-2024


icp4_neural.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Gemini
Epoch 1/5
1563/1563 [==============================] - 16s 9ms/step - loss: 1.8508 - accuracy: 0.3115 - val_loss: 1.5590 - val_
Epoch 2/5
1563/1563 [==============================] - 16s 10ms/step - loss: 1.4047 - accuracy: 0.4875 - val_loss: 1.2387 - val_accuracy: 0.5526
Epoch 3/5
1563/1563 [==============================] - 13s 8ms/step - loss: 1.1951 - accuracy: 0.5733 - val_loss: 1.1510 - val_accuracy: 0.5880
Epoch 4/5
1563/1563 [==============================] - 13s 8ms/step - loss: 1.0507 - accuracy: 0.6233 - val_loss: 0.9887 - val_accuracy: 0.6514
Epoch 5/5
1563/1563 [==============================] - 15s 9ms/step - loss: 0.9474 - accuracy: 0.6632 - val_loss: 0.9493 - val_accuracy: 0.6669
Accuracy: 66.69%
1/1 [==============================] - 0s 141ms/step
Predicted classes: [3 8 8 0]
Actual classes: [3 8 8 0]
Pred: 3, Actual: 3
Pred: 8, Actual: 8
Pred: 8, Actual: 8
Pred: 0, Actual: 0
Executing (7m 45s)
SENSEX -0.03%
Search
ENG IN
10:57 11-07-2024


Content
15426092
icp4_neural.ipynb
Home - Google Dr
icp4_neural_assign
kartikanumolu1/nc
colab.research.google.com/drive/10I9VaS4rb2y9eOIT7K0qODarhXMLZBjm#scrollTo=iC4fLy1T-W9t
icp4_neural.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
T4
RAM
Disk
Gemini
Model Loss
train_loss
val_loss
Loss
Epoch
Model Accuracy
train_accuracy
val_accuracy
Accuracy
Epoch
Model Loss
Model Accuracy
Executing (7m 53s)
SENSEX
-0.03%
Search
ENG
IN
10:57
11-07-2024

Content
15426092
icp4_neural.ipynb
Home - Google Dr
icp4_neural_assign
kartikanumolu1/nc
colab.research.google.com/drive/10I9VaS4rb2y9eOIT7K0qODarhXMLZBjm#scrollTo=iC4fLy1T-W9t
icp4_neural.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
T4
RAM
Disk
Gemini
Model Loss
train_loss
val_loss
Loss
Epoch
Model Accuracy
train_accuracy
val_accuracy
Accuracy
Epoch
Executing (7m 57s)
SENSEX
-0.03%
Search
ENG
IN
10:57
11-07-2024