Neural Networks & Deep Learning: ICP3

Name: Ganesh Kumar Kokkera
ID:700743028

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?

```
import numpy as np
from keras.datasets import cifar10
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.constraints import maxnorm
from keras.utils import np_utils
from keras.optimizers import SGD

# 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 = np_utils.to_categorical(y_train)
y_test = np_utils.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))
```

Neural Networks & Deep Learning: ICP3

Name: Ganesh Kumar Kokkera
ID:700743028

```
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)))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))

# Compile model
epochs = 5
learning_rate = 0.01
decay_rate = learning_rate / epochs
sgd = SGD(lr=learning_rate, momentum=0.9, decay=decay_rate)
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)

# Evaluate 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])
# Convert the predictions to class labels
predicted_labels = np.argmax(predictions, axis=1)
# Convert the actual labels to class labels
actual_labels = np.argmax(y_test[:4], axis=1)
```

Neural Networks & Deep Learning: ICP3
Name: Ganesh Kumar Kokkera
ID:700743028

Did the performance change?

The performance of the model is likely to improve with the addition of more layers and higher number of feature maps, but it will also increase the complexity and the training time of the model. The new model architecture provided in the instruction includes several new layers and higher number of feature maps, which may improve the accuracy of the model.

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_24 (Conv2D)          (None, 32, 32, 32)        896       
                                                                 
 dropout_24 (Dropout)        (None, 32, 32, 32)        0         
                                                                 
 conv2d_25 (Conv2D)          (None, 32, 32, 32)        9248      
                                                                 
 max_pooling2d_12 (MaxPoolin  (None, 16, 16, 32)       0         
 g2D)                                                            
                                                                 
 conv2d_26 (Conv2D)          (None, 16, 16, 64)        18496     
                                                                 
 dropout_25 (Dropout)        (None, 16, 16, 64)        0         
                                                                 
 conv2d_27 (Conv2D)          (None, 16, 16, 64)        36928     
                                                                 
 max_pooling2d_13 (MaxPoolin  (None, 8, 8, 64)         0         
 g2D)                                                            
                                                                 
 conv2d_28 (Conv2D)          (None, 8, 8, 128)         73856     
                                                                 
 dropout_26 (Dropout)        (None, 8, 8, 128)         0         
                                                                 
 conv2d_29 (Conv2D)          (None, 8, 8, 128)         147584    
                                                                 
 max_pooling2d_14 (MaxPoolin  (None, 4, 4, 128)        0         
 g2D)                                                            
                                                                 
 flatten_4 (Flatten)         (None, 2048)              0         
                                                                 
 dropout_27 (Dropout)        (None, 2048)              0         
```

Neural Networks & Deep Learning: ICP3

Name: Ganesh Kumar Kokkera
ID:700743028

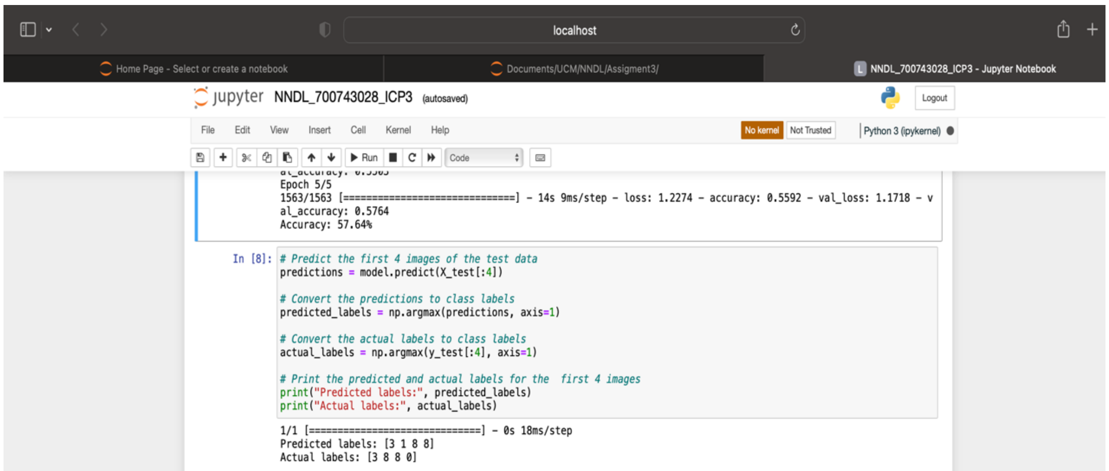
```
 dense_12 (Dense)            (None, 1024)              2098176

 dropout_28 (Dropout)        (None, 1024)              0

 dense_13 (Dense)            (None, 512)               524800

 dropout_29 (Dropout)        (None, 512)               0

 dense_14 (Dense)            (None, 10)                5130

=================================================================
Total params: 2,915,114
Trainable params: 2,915,114
Non-trainable params: 0
_________________________________________________________________
/usr/local/lib/python3.10/dist-packages/keras/optimizers/legacy/gradient_descent.py:114: UserWarning: The `lr` arg
ument is deprecated, use `learning_rate` instead.
  super().__init__(name, **kwargs)
None
Epoch 1/5
1563/1563 [==============================] - 18s 10ms/step - loss: 1.8726 - accuracy: 0.3072 - val_loss: 1.6348 -
val_accuracy: 0.4091
Epoch 2/5
1563/1563 [==============================] - 12s 8ms/step - loss: 1.4955 - accuracy: 0.4540 - val_loss: 1.4204 - v
al_accuracy: 0.4952
Epoch 3/5
1563/1563 [==============================] - 12s 8ms/step - loss: 1.3675 - accuracy: 0.5027 - val_loss: 1.3329 - v
al_accuracy: 0.5164
Epoch 4/5
1563/1563 [==============================] - 12s 8ms/step - loss: 1.2868 - accuracy: 0.5360 - val_loss: 1.2500 - v
al_accuracy: 0.5503
Epoch 5/5
1563/1563 [==============================] - 14s 9ms/step - loss: 1.2274 - accuracy: 0.5592 - val_loss: 1.1718 - v
al_accuracy: 0.5764
Accuracy: 57.64%
```

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.

```
# Print the predicted and actual labels for the first 4 images
print("Predicted labels:", predicted_labels)
print("Actual labels:   ", actual_label

import matplotlib.pyplot as plt


# Plot the training and validation loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
```

Neural Networks & Deep Learning: ICP3
Name: Ganesh Kumar Kokkera
ID:700743028

```
plt.legend(['train', 'val'], loc='upper right')

plt.show()
```



Neural Networks & Deep Learning: ICP3

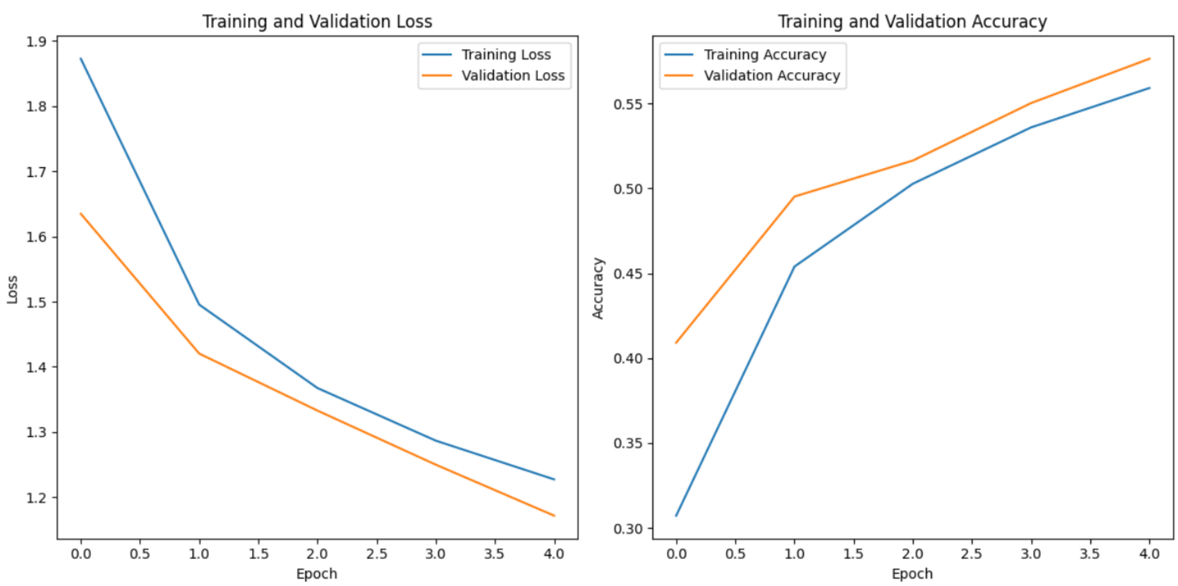
Name: Ganesh Kumar Kokkera
ID:700743028

3. Visualize Loss and Accuracy using the history object.

```
import matplotlib.pyplot as plt

# Plot the training and validation loss
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Training and Validation Loss')
plt.legend()

# Plot the training and validation accuracy
plt.subplot(1, 2, 2)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.title('Training and Validation Accuracy')
plt.legend()

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

Neural Networks & Deep Learning: ICP3
Name: Ganesh Kumar Kokkera
ID:700743028

In [11]:
```
import matplotlib.pyplot as plt

# Plot the training and validation loss
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Training and Validation Loss')
plt.legend()

# Plot the training and validation accuracy
plt.subplot(1, 2, 2)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.title('Training and Validation Accuracy')
plt.legend()

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



GIT HUB LINK :
https://github.com/gxk30280/700743028_NNDL_Assignment3.git

VIDEO LINK:
https://drive.google.com/file/d/1SCupRmQmmQqNJp-hnbhLgFnXMs2LAiHd/view?usp=sharing