```
import tensorflow as tf from tensorflow.keras import datasets, layers, models import
matplotlib.pyplot as plt # Load and preprocess the CIFAR-10 dataset (train_images,
train_labels), (test_images, test_labels) = datasets.cifar10.load_data() # Normalize pixel values
to be between 0 and 1 train_images, test_images = train_images / 255.0, test_images / 255.0 #
Data Augmentation data_augmentation = tf.keras.Sequential([
layers.experimental.preprocessing.RandomFlip("horizontal"),
layers.experimental.preprocessing.RandomRotation(0.1), ]) # Build the CNN model model =
models.Sequential([ data_augmentation, layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(32, 32, 3)), layers.MaxPooling2D((2, 2)), layers.Conv2D(64, (3, 3),
activation='relu'), layers.MaxPooling2D((2, 2)), layers.Conv2D(64, (3, 3), activation='relu'),
layers.Flatten(), layers.Dense(64, activation='relu'), layers.Dropout(0.5), layers.Dense(10) ]) #
Compile the model model.compile(optimizer='adam',
loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True), metrics=['accuracy']) #
Train the model history = model.fit(train_images, train_labels, epochs=10, validation_data=
(test_images, test_labels)) # Evaluate the model test_loss, test_acc =
model.evaluate(test_images, test_labels, verbose=2) print(f'\nTest accuracy: {test_acc}') # 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', 'Validation'], 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', 'Validation'],
loc='upper left') plt.show()
```