```
 1: import tensorflow as tf
 2: from tensorflow import keras
 3: from tensorflow.keras import layers, regularizers
 4: from keras.layers import Dense, Dropout, Activation, Flatten, BatchNormalization
 5: from keras.layers import Conv2D, MaxPooling2D, LeakyReLU
 6: from sklearn.metrics import confusion_matrix, classification_report
 7: from sklearn.utils import shuffle
 8: import matplotlib.pyplot as plt
 9: plt.rc('font', size=18)
10: plt.rcParams['figure.constrained_layout.use'] = True
11: import sys
12: import even_samples
13: import keras
14: import math
15:
16: (x_train, y_train), (x_test, y_test) = keras.datasets.cifar10.load_data()
17: # x_train, y_train = even_samples.even_sample_categories(num_samples_per_class=math.floor(100/10))
18: n=5000
19: x_train = x_train[1:n]; y_train=y_train[1:n]
20: num_classes = 10
21: input_shape = (32, 32, 3)
22:
23: # Scale images to the [0, 1] range
24: x_train = x_train.astype("float32") / 255
25: x_test = x_test.astype("float32") / 255
26:
27: # convert class vectors to binary class matrices
28: y_train = keras.utils.to_categorical(y_train, num_classes)
29: y_test = keras.utils.to_categorical(y_test, num_classes)
30:
31: model = keras.Sequential()
32: model.add(Conv2D(16, (3,3), padding='same', input_shape=x_train.shape[1:],activation='relu'))
33: model.add(Conv2D(16, (3,3), strides=(2,2), padding='same', activation='relu'))
34: model.add(Conv2D(32, (3,3), padding='same', activation='relu'))
35: model.add(Conv2D(32, (3,3), strides=(2,2), padding='same', activation='relu'))
36: model.add(Dropout(0.5))
37: model.add(Flatten())
38: model.add(Dense(num_classes, activation='softmax',kernel_regularizer=regularizers.l1(0.0001)))
39: model.compile(loss="categorical_crossentropy", optimizer='adam', metrics=["accuracy"])
40: model.summary()
41:
42: batch_size = 128
43: epochs = 20
44: history = model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs, validation_split=0.1)
45: model.save("cifar.model")
46: plt.subplot(211)
47: plt.plot(history.history['accuracy'])
48: plt.plot(history.history['val_accuracy'])
49: plt.title('model accuracy')
50: plt.ylabel('accuracy')
51: plt.xlabel('epoch')
52: plt.legend(['train', 'val'], loc='upper left')
53: plt.subplot(212)
54: plt.plot(history.history['loss'])
55: plt.plot(history.history['val_loss'])
56: plt.title('model loss')
57: plt.ylabel('loss'); plt.xlabel('epoch')
58: plt.legend(['train', 'val'], loc='upper left')
59: plt.show()
```