untitled5

November 19, 2023

[1]:
```
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import random
import pandas as pd
import numpy as np

(train_images, train_labels), (test_images, test_labels) = datasets.cifar10.
  ↪load_data()
train_images, test_images = train_images / 255.0, test_images / 255.0

type(train_images)
```

[1]:
```
numpy.ndarray
```

[2]:
```
class_names = ['airplane', 'automobile', 'bird', 'cat', 'deer','dog', 'frog',␣
  ↪'horse', 'ship', 'truck']

plt.figure(figsize=(10,10))
for i in range(10):
    plt.subplot(5,5,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(train_images[i])
    plt.xlabel(class_names[train_labels[i][0]])
plt.show()
```

[3]:
```
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10))

model.summary()
```

```
Model: "sequential"

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 30, 30, 32)        896

 max_pooling2d (MaxPooling2  (None, 15, 15, 32)        0
 D)

 conv2d_1 (Conv2D)           (None, 13, 13, 64)        18496

 max_pooling2d_1 (MaxPoolin  (None, 6, 6, 64)          0
 g2D)

 conv2d_2 (Conv2D)           (None, 4, 4, 64)          36928

 flatten (Flatten)           (None, 1024)              0

 dense (Dense)               (None, 64)                65600
 dense_1 (Dense)             (None, 10)                650

=================================================================
Total params: 122570 (478.79 KB)
Trainable params: 122570 (478.79 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

[4]:
```
model.compile(optimizer='adam', loss=tf.keras.losses.
 ↪SparseCategoricalCrossentropy(from_logits=True), metrics=['accuracy'])

epochs = 2
h = model.fit(train_images, train_labels, epochs=epochs,␣
 ↪validation_data=(test_images, test_labels))
```

```
Epoch 1/2
1563/1563 [==============================] - 42s 26ms/step - loss: 1.4974 -
accuracy: 0.4555 - val_loss: 1.2046 - val_accuracy: 0.5642
Epoch 2/2
1563/1563 [==============================] - 38s 24ms/step - loss: 1.1259 -
accuracy: 0.6024 - val_loss: 1.0424 - val_accuracy: 0.6347
```

[5]:
```
predicted_values = model.predict(test_images)
predicted_values.shape

n = random.randint(0, 9999)
plt.figure(figsize=(10, 10))
plt.imshow(test_images[n])
plt.xticks([])
plt.yticks([])
plt.grid(False)
plt.title(class_names[np.argmax(predicted_values[n])])

test_loss, test_acc = model.evaluate(test_images, test_labels)
print("loss %.3f" % test_loss)
print("acc %.3f" % test_acc)
```

```
313/313 [==============================] - 2s 6ms/step
313/313 [==============================] - 2s 6ms/step - loss: 1.0424 -
accuracy: 0.6347
loss 1.042
acc 0.635
```

dog

```
[ ]:
```