CNN for CIFAR10 with Tensorflow 2[¶](#gjdgxs)

Import TensorFlow[¶](#30j0zll)

In [1]:

*# TensorFlow and tf.keras*  
**import** tensorflow **as** tf  
*# from tensorflow.keras import datasets, layers, models*  
  
*# Helper libraries*  
**import** matplotlib.pyplot **as** plt  
**import** numpy **as** np  
  
print(tf**.**\_\_version\_\_)

2.4.1

Download and prepare the CIFAR-10 dataset[¶](#1fob9te)

In [2]:

*# Import and load the CIFAR-10 data directly from tf.keras*  
(train\_images, train\_labels), (test\_images, test\_labels) **=** tf**.**keras**.**datasets**.**cifar10**.**load\_data()

In [3]:

class\_names **=** ['Airplane', 'Automobile', 'Bird', 'Cat', 'Deer',  
 'Dog', 'Frog', 'Horse', 'Ship', 'Truck']

Explore the data[¶](#3znysh7)

In [4]:

train\_images**.**shape

Out[4]:

(50000, 32, 32, 3)

In [5]:

len(train\_labels)

Out[5]:

50000

In [6]:

train\_labels

Out[6]:

array([[6],  
 [9],  
 [9],  
 ...,  
 [9],  
 [1],  
 [1]], dtype=uint8)

In [7]:

test\_images**.**shape

Out[7]:

(10000, 32, 32, 3)

In [8]:

len(test\_labels)

Out[8]:

10000

Preprocess the data[¶](#2et92p0)

In [9]:

train\_images, test\_images **=** train\_images **/** 255.0, test\_images **/** 255.0

Build the model[¶](#tyjcwt)

Set up parameters[¶](#3dy6vkm)

In [10]:

param\_epoch\_count **=** 20  
param\_batch\_size **=** 32  
param\_act\_fn **=** 'relu'  
param\_optimizer **=** 'adam'  
param\_loss\_fn **=** tf**.**keras**.**losses**.**SparseCategoricalCrossentropy(from\_logits**=True**)

Set up architecture[¶](#1t3h5sf)

In [11]:

model **=** tf**.**keras**.**Sequential()  
  
*#### Create the convolutional base*  
model**.**add(tf**.**keras**.**layers**.**Conv2D(32, (3, 3), activation**=**param\_act\_fn, input\_shape**=**(32, 32, 3)))  
model**.**add(tf**.**keras**.**layers**.**MaxPooling2D((2, 2)))  
model**.**add(tf**.**keras**.**layers**.**Conv2D(64, (3, 3), activation**=**param\_act\_fn))  
model**.**add(tf**.**keras**.**layers**.**MaxPooling2D((2, 2)))  
model**.**add(tf**.**keras**.**layers**.**Conv2D(64, (3, 3), activation**=**param\_act\_fn))  
  
*#### Add Dense layers on top*  
model**.**add(tf**.**keras**.**layers**.**Flatten())  
model**.**add(tf**.**keras**.**layers**.**Dense(64, activation**=**param\_act\_fn))  
model**.**add(tf**.**keras**.**layers**.**Dense(10))

In [12]:

model**.**summary()

Model: "sequential"  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Layer (type) Output Shape Param #   
=================================================================  
conv2d (Conv2D) (None, 30, 30, 32) 896   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
max\_pooling2d (MaxPooling2D) (None, 15, 15, 32) 0   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
conv2d\_1 (Conv2D) (None, 13, 13, 64) 18496   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
max\_pooling2d\_1 (MaxPooling2 (None, 6, 6, 64) 0   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
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: 122,570  
Trainable params: 122,570  
Non-trainable params: 0  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Compile the model[¶](#4d34og8)

In [13]:

model**.**compile(optimizer**=**param\_optimizer,  
 loss**=**param\_loss\_fn,  
 metrics**=**['accuracy'])

Train the model[¶](#2s8eyo1)

In [14]:

history **=** model**.**fit(train\_images, train\_labels, batch\_size**=**32, epochs**=**param\_epoch\_count, validation\_data**=**(test\_images, test\_labels))

Epoch 1/20  
1563/1563 [==============================] - 10s 5ms/step - loss: 1.7880 - accuracy: 0.3366 - val\_loss: 1.2610 - val\_accuracy: 0.5515  
Epoch 2/20  
1563/1563 [==============================] - 6s 4ms/step - loss: 1.2029 - accuracy: 0.5720 - val\_loss: 1.1205 - val\_accuracy: 0.6004  
Epoch 3/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 1.0284 - accuracy: 0.6396 - val\_loss: 1.1308 - val\_accuracy: 0.5980  
Epoch 4/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.9347 - accuracy: 0.6714 - val\_loss: 0.9813 - val\_accuracy: 0.6534  
Epoch 5/20  
1563/1563 [==============================] - 6s 4ms/step - loss: 0.8393 - accuracy: 0.7060 - val\_loss: 0.9205 - val\_accuracy: 0.6776  
Epoch 6/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.7863 - accuracy: 0.7228 - val\_loss: 0.8681 - val\_accuracy: 0.6915  
Epoch 7/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.7434 - accuracy: 0.7378 - val\_loss: 0.8796 - val\_accuracy: 0.7000  
Epoch 8/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.6895 - accuracy: 0.7590 - val\_loss: 0.8425 - val\_accuracy: 0.7123  
Epoch 9/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.6442 - accuracy: 0.7745 - val\_loss: 0.8834 - val\_accuracy: 0.7006  
Epoch 10/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.6098 - accuracy: 0.7853 - val\_loss: 0.9849 - val\_accuracy: 0.6832  
Epoch 11/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.5808 - accuracy: 0.7946 - val\_loss: 0.8866 - val\_accuracy: 0.7071  
Epoch 12/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.5343 - accuracy: 0.8119 - val\_loss: 0.9037 - val\_accuracy: 0.7011  
Epoch 13/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.5143 - accuracy: 0.8192 - val\_loss: 0.9136 - val\_accuracy: 0.7098  
Epoch 14/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.4843 - accuracy: 0.8299 - val\_loss: 0.9386 - val\_accuracy: 0.7079  
Epoch 15/20  
1563/1563 [==============================] - 6s 4ms/step - loss: 0.4499 - accuracy: 0.8417 - val\_loss: 0.9476 - val\_accuracy: 0.7090  
Epoch 16/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.4225 - accuracy: 0.8488 - val\_loss: 0.9687 - val\_accuracy: 0.7073  
Epoch 17/20  
1563/1563 [==============================] - 6s 4ms/step - loss: 0.4068 - accuracy: 0.8562 - val\_loss: 1.0367 - val\_accuracy: 0.6992  
Epoch 18/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.3686 - accuracy: 0.8697 - val\_loss: 1.0328 - val\_accuracy: 0.7099  
Epoch 19/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.3538 - accuracy: 0.8733 - val\_loss: 1.0831 - val\_accuracy: 0.7040  
Epoch 20/20  
1563/1563 [==============================] - 7s 4ms/step - loss: 0.3331 - accuracy: 0.8817 - val\_loss: 1.1210 - val\_accuracy: 0.6988

Evaluate the model[¶](#17dp8vu)

In [20]:

fig, ax **=** plt**.**subplots()  
fig**.**set\_size\_inches(18.5, 10.5)  
  
ax**.**xaxis**.**set\_ticks(np**.**arange(0, param\_epoch\_count**+**1))  
ax**.**yaxis**.**set\_ticks(np**.**arange(0, 1.1, 0.1))  
ax**.**grid()  
  
train\_acc **=** history**.**history['accuracy']  
test\_acc **=** history**.**history['val\_accuracy']  
  
color\_train**=**'#238ef0'  
color\_test**=**'#01b1b3'  
  
plt**.**rcParams**.**update({'font.size': 18})  
  
plt**.**plot(np**.**arange(1,param\_epoch\_count**+**1), train\_acc, label**=**'Mokymo', c**=**color\_train, linewidth**=**4)  
plt**.**plot(np**.**arange(1,param\_epoch\_count**+**1), test\_acc, label**=**'Testavimo', c**=**color\_test, linewidth**=**4)  
  
plt**.**plot(param\_epoch\_count, train\_acc[**-**1],'co', c**=**color\_train, markersize**=**15)  
plt**.**text(param\_epoch\_count, train\_acc[**-**1], "{:.4f}"**.**format(train\_acc[**-**1]), c**=**color\_train)  
  
plt**.**plot(param\_epoch\_count, test\_acc[**-**1], 'co', c**=**color\_test, markersize**=**15)  
plt**.**text(param\_epoch\_count, test\_acc[**-**1], "{:.4f}"**.**format(test\_acc[**-**1]), c**=**color\_test)  
  
plt**.**xlabel('Epochos', fontsize**=**18)  
plt**.**ylabel('Klasifikavimo tikslumas', fontsize**=**18)  
  
plt**.**xlim([1, param\_epoch\_count])  
plt**.**ylim([0.0, 1.0])  
  
plt**.**legend(loc**=**'lower right')

Out[20]:

<matplotlib.legend.Legend at 0x7f01776b5b70>