```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# Define the model architecture
model = keras.Sequential([
    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(128, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(10, activation='softmax')
])

# Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Load the dataset
from tensorflow.keras.datasets import cifar10

(x_train, y_train), (x_test, y_test) = cifar10.load_data()

# Preprocess the data
x_train = x_train / 255.0
x_test = x_test / 255.0
y_train = keras.utils.to_categorical(y_train, 10) # one hot encoding
y_test = keras.utils.to_categorical(y_test, 10)

# Train the model
model.fit(x_train, y_train, epochs=10, validation_data=(x_test, y_test))

# Evaluate the model
test_loss, test_acc = model.evaluate(x_test, y_test, verbose=2)
print('Test accuracy:', test_acc)
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 [==============================] - 6s 0us/step
Epoch 1/10
1563/1563 [==============================] - 68s 43ms/step - loss: 1.4965 - accuracy: 0.4522 - val_loss: 1.2363 - val_acc
Epoch 2/10
1563/1563 [==============================] - 65s 42ms/step - loss: 1.1153 - accuracy: 0.6055 - val_loss: 1.0530 - val_acc
Epoch 3/10
1563/1563 [==============================] - 65s 42ms/step - loss: 0.9703 - accuracy: 0.6602 - val_loss: 0.9607 - val_acc
Epoch 4/10
1563/1563 [==============================] - 64s 41ms/step - loss: 0.8579 - accuracy: 0.7000 - val_loss: 0.9097 - val_acc
Epoch 5/10
1563/1563 [==============================] - 64s 41ms/step - loss: 0.7766 - accuracy: 0.7296 - val_loss: 0.8770 - val_acc
Epoch 6/10
1563/1563 [==============================] - 64s 41ms/step - loss: 0.7053 - accuracy: 0.7522 - val_loss: 0.8754 - val_acc
Epoch 7/10
1563/1563 [==============================] - 63s 41ms/step - loss: 0.6494 - accuracy: 0.7713 - val_loss: 0.9005 - val_acc
Epoch 8/10
1563/1563 [==============================] - 64s 41ms/step - loss: 0.5955 - accuracy: 0.7902 - val_loss: 0.8580 - val_acc
Epoch 9/10
1563/1563 [==============================] - 64s 41ms/step - loss: 0.5459 - accuracy: 0.8072 - val_loss: 0.8850 - val_acc
Epoch 10/10
1563/1563 [==============================] - 65s 41ms/step - loss: 0.5006 - accuracy: 0.8239 - val_loss: 0.8974 - val_acc
313/313 - 3s - loss: 0.8974 - accuracy: 0.7095 - 3s/epoch - 10ms/step
Test accuracy: 0.7095000147819519
```

```
# Load a test image and preprocess it
import cv2
import numpy as np
img = cv2.imread('/content/drive/MyDrive/Colab Notebooks/cats-dogs-small/test/cats/cat.601.jpg')
img = cv2.resize(img, (32, 32))
img = np.expand_dims(img, axis=0)
img = img / 255.0
# Predict the label for the test image
prediction = model.predict(img)
label = np.argmax(prediction)
print('Predicted label:', label)
```

```
1/1 [==============================] - 0s 19ms/step
Predicted label: 3
```

```
#airplane : 0
#automobile : 1
```

```
#bird : 2
#cat : 3
#deer : 4
#dog : 5
#frog : 6
#horse : 7
#ship : 8
#truck : 9
```

```

```