```python
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
import numpy as np
import matplotlib.pyplot as plt

# Load CIFAR-10 dataset
(x_train, y_train), (x_test, y_test) = keras.datasets.cifar10.load_data()

# Normalize the pixel values to the range [0, 1]
x_train = x_train.astype("float32") / 255.0
x_test = x_test.astype("float32") / 255.0

# Flatten the images for a fully connected network
x_train = x_train.reshape(x_train.shape[0], -1)
x_test = x_test.reshape(x_test.shape[0], -1)

# Convert class labels to one-hot encoding
y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)

# Define the fully connected neural network model
model = keras.Sequential([
    layers.Dense(1024, activation='relu', input_shape=(3072,)),
    layers.Dropout(0.3),
    layers.Dense(512, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(256, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(128, activation='relu'),
    layers.Dense(10, activation='softmax')
])

# Compile the model
model.compile(optimizer=keras.optimizers.Adam(learning_rate=0.001),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Train the model
history = model.fit(x_train, y_train, epochs=5, batch_size=64, 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)

# Plot training history
plt.plot(history.history['accuracy'], label='train accuracy')
plt.plot(history.history['val_accuracy'], label='validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argumen
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/5
782/782 ━━━━━━━━━━━━━━━━━━━━ 60s 74ms/step - accuracy: 0.1884 - loss: 2.1794 - val_accuracy: 0.3094 - val_loss: 1.8956
Epoch 2/5
782/782 ━━━━━━━━━━━━━━━━━━━━ 77s 67ms/step - accuracy: 0.2817 - loss: 1.9363 - val_accuracy: 0.3334 - val_loss: 1.8247
Epoch 3/5
782/782 ━━━━━━━━━━━━━━━━━━━━ 85s 71ms/step - accuracy: 0.3014 - loss: 1.8916 - val_accuracy: 0.3458 - val_loss: 1.8219
Epoch 4/5
782/782 ━━━━━━━━━━━━━━━━━━━━ 79s 68ms/step - accuracy: 0.3060 - loss: 1.8742 - val_accuracy: 0.3635 - val_loss: 1.7876
Epoch 5/5
782/782 ━━━━━━━━━━━━━━━━━━━━ 82s 68ms/step - accuracy: 0.3200 - loss: 1.8433 - val_accuracy: 0.3479 - val_loss: 1.7965
313/313 - 3s - 10ms/step - accuracy: 0.3479 - loss: 1.7965
Test accuracy: 0.34790000319480896
```