TUTORIAL

AI - Apple Silicon Mac M1 机器学习环境 (TensorFlow, JupyterLab, VSCode)

ai deep learning machine learning m1 mac apple silicon 人工智能 机器学习 深度学习 tensorflow jupyter vscode

Read in English

> *注：由于 https://github.com/apple/tensorflow_macos 已经 archived，建议大家根据 Apple Silicon Mac M1 机器学习环境 (tensorflow-metal PluggableDevice, JupyterLab, VSCode) 安装最新支持 GPU 加速的 TensorFlow。*

Xcode

从 App Store 安装 Xcode。

Command Line Tools

从 Apple Developer 下载安装 Xcode Command Line Tools 或者执行以下命令。

```
$ xcode-select --install
```

Homebrew

```
$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

Miniforge

> *Anaconda 无法在 M1 上运行, Miniforge 是用来替代它的。*

从 https://github.com/conda-forge/miniforge 下载 `Miniforge3-MacOSX-arm64` 。

Miniforge3

Latest installers with Python 3.8 (*) in the base environment:

OS	Architecture	Download
Linux	x86_64 (amd64)	Miniforge3-Linux-x86_64
Linux	aarch64 (arm64)	Miniforge3-Linux-aarch64
Linux	ppc64le (POWER8/9)	Miniforge3-Linux-ppc64le
OS X	x86_64	Miniforge3-MacOSX-x86_64
OS X	arm64 (Apple Silicon) (**)	Miniforge3-MacOSX-arm64
Windows	x86_64	Miniforge3-Windows-x86_64

执行以下命令，安装 `Miniforge`

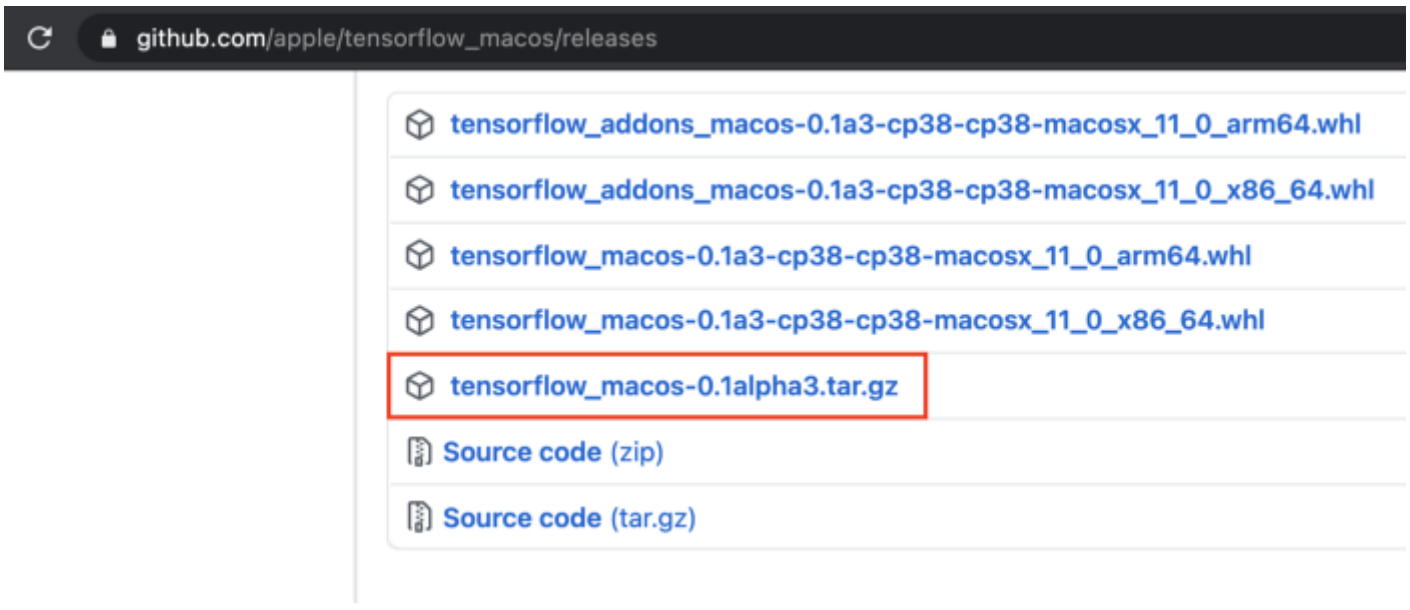
```
$ bash Miniforge3-MacOSX-arm64.sh
```

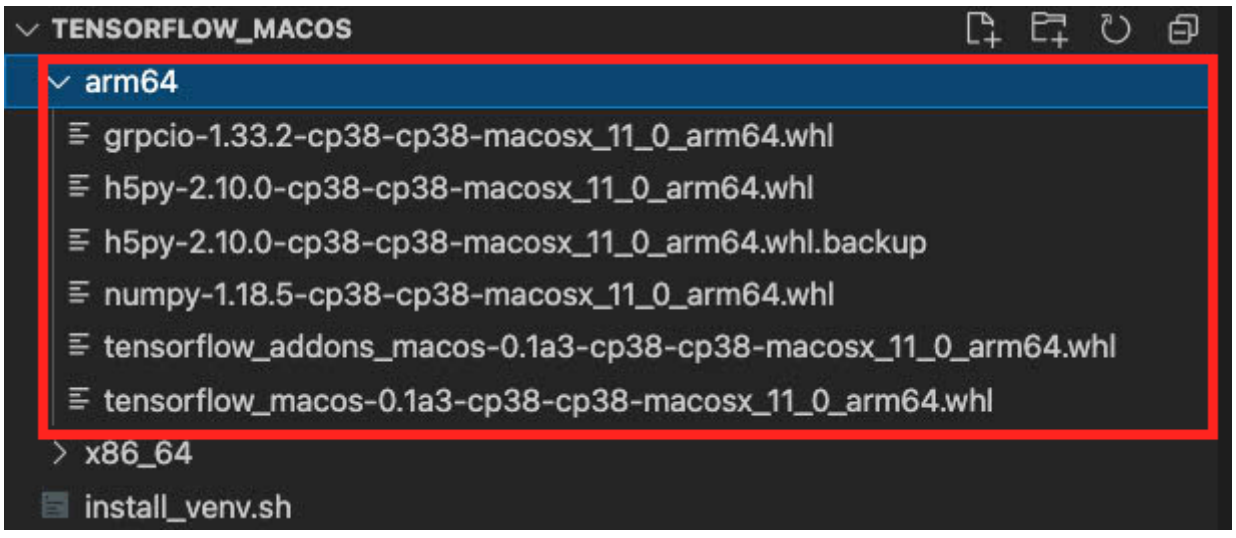
重启终端并检查 Python 安装情况。

```
$ which python
/Users/catchzeng/miniforge3/bin/python
$ which pip
/Users/catchzeng/miniforge3/bin/pip
```

下载 Apple TensorFlow

从 https://github.com/apple/tensorflow_macos/releases 下载 TensorFlow 并解压，然后进入 `arm64` 目录下。





创建虚拟环境

创建一个 conda 创建虚拟环境，这里使用 `python 3.8` (ATF 2.4 需要)。

```
$ conda create -n tensorflow python=3.8
$ conda activate tensorflow
```

安装必须的包

```
$ brew install libjpeg
$ conda install -y pandas matplotlib scikit-learn jupyterlab
```

> *注意: libjpeg 是 matplotlib 需要依赖的库。*

安装特殊版本的 pip 和其他包

```
$ pip install --force pip==20.2.4 wheel setuptools cached-property six packaging
```

> *注意: Apple TensorFlow 特殊版本的 pip。*

安装 Apple 提供的包(numpy, grpcio, h5py)

```
$ pip install --upgrade --no-dependencies --force numpy-1.18.5-cp38-cp38-macosx_11_0_arm64.whl grpcio-1.33.2-cp38-cp38-macosx_11_0_arm64.whl h5py-2.10.0-cp38-cp38-macosx_11_0_arm64.whl
```

安装额外的包

```
$ pip install absl-py astunparse flatbuffers gast google_pasta keras_preprocessing opt_einsum protobuf tensorflow_estimator termcolor typing_extensions wrapt wheel tensorboard typeguard
```

安装 TensorFlow

```
$ pip install --upgrade --no-dependencies --force tensorflow_macos-0.1a3-cp38-cp38-macosx_11_0_arm64.whl
$ pip install --upgrade --no-dependencies --force tensorflow_addons_macos-0.1a3-cp38-cp38-macosx_11_0_arm64.whl
```

最后，升级 pip 到正确的版本。

```
$ pip install --upgrade pip
```

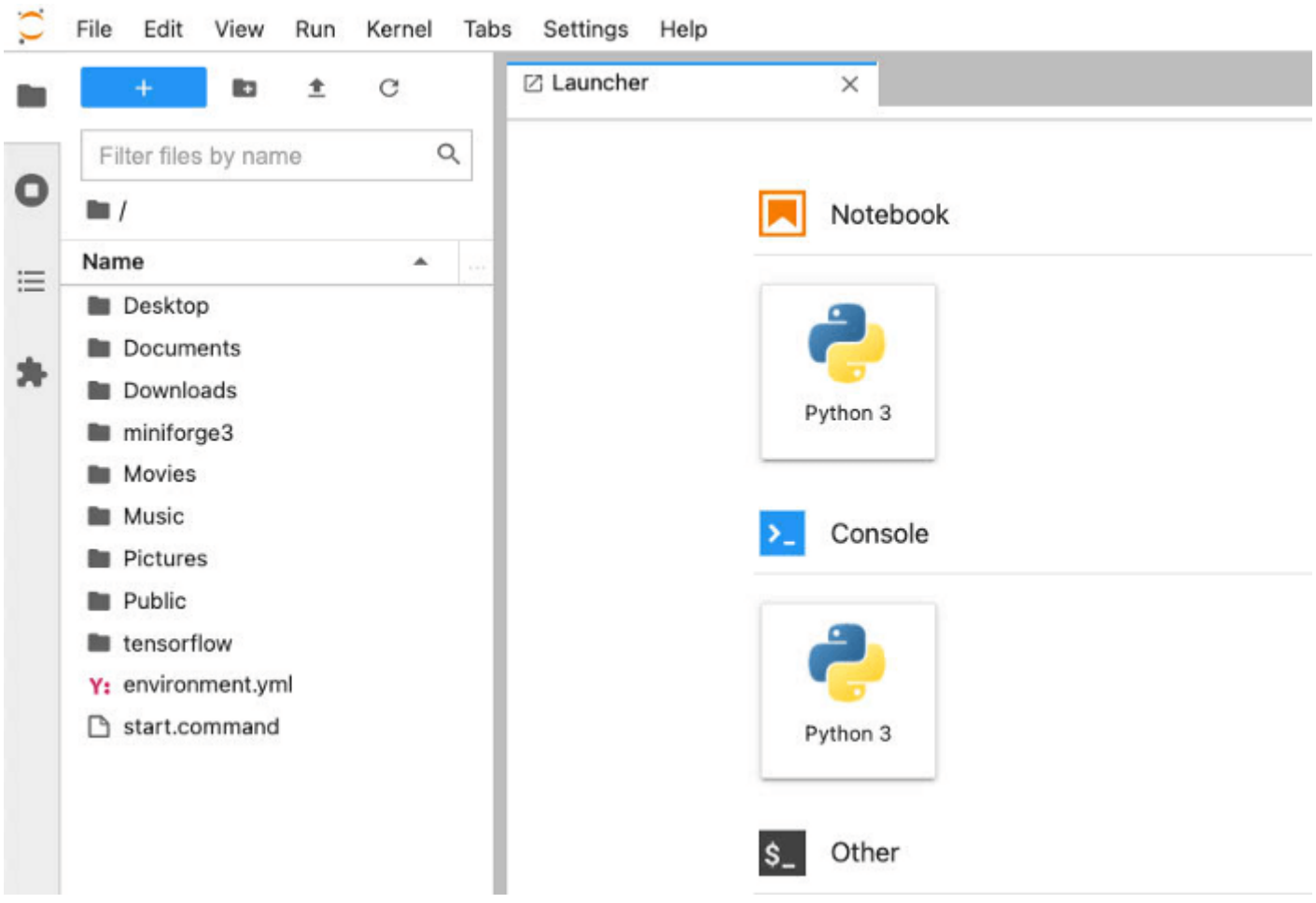
测试

TensorFlow

```
$ python
Python 3.8.8 | packaged by conda-forge | (default, Feb 20 2021, 15:50:57)
[Clang 11.0.1 ] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow as tf
>>> print(tf.__version__)
2.4.0-rc0
>>>
```

JupyterLab

```
$ jupyter lab
```




```
from tensorflow.keras import layers
from tensorflow.keras import models
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
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, activation='softmax'))
model.summary()
```

[1]:
```
from tensorflow.keras import layers
from tensorflow.keras import models

model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
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, activation='softmax'))

model.summary()
```

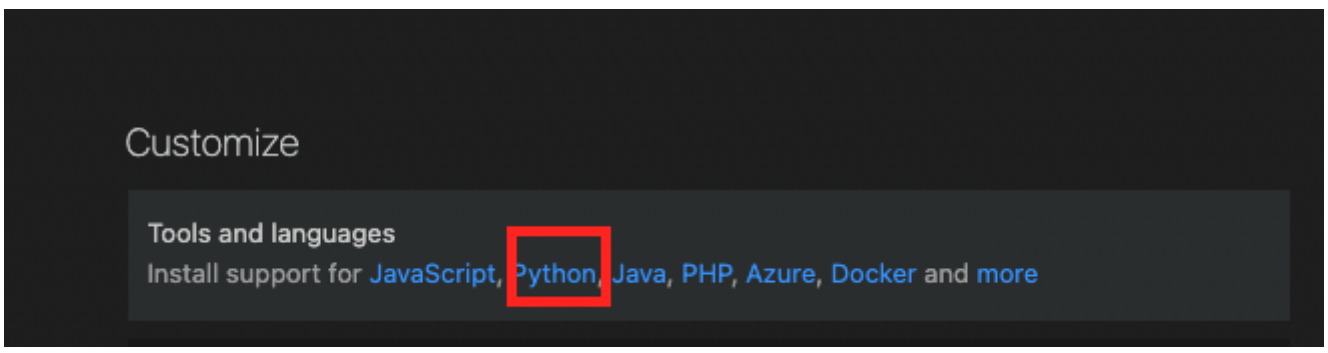
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 3, 3, 64)          36928
_________________________________________________________________
flatten (Flatten)            (None, 576)               0
_________________________________________________________________
dense (Dense)                (None, 64)                36928
_________________________________________________________________
dense_1 (Dense)              (None, 10)                650
=================================================================
Total params: 93,322
Trainable params: 93,322
Non-trainable params: 0
_________________________________________________________________
```

```
from tensorflow.keras.datasets import mnist
from tensorflow.keras.utils import to_categorical
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
train_images = train_images.reshape((60000, 28, 28, 1))
train_images = train_images.astype('float32') / 255
test_images = test_images.reshape((10000, 28, 28, 1))
test_images = test_images.astype('float32') / 255
train_labels = to_categorical(train_labels)
test_labels = to_categorical(test_labels)
model.compile(optimizer='rmsprop',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
model.fit(train_images, train_labels, epochs=5, batch_size=64)
test_loss, test_acc = model.evaluate(test_images, test_labels)
test_acc
```

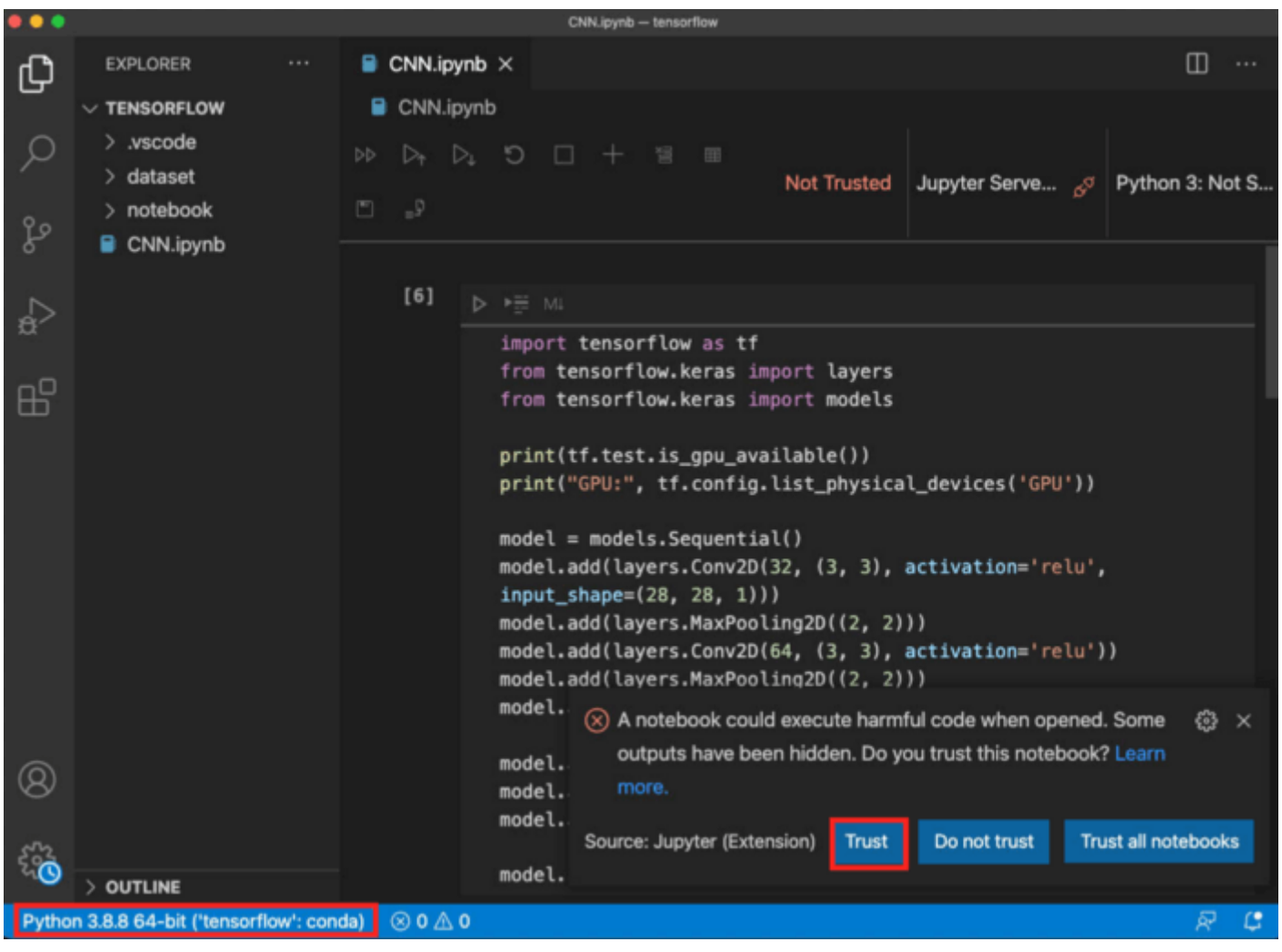
```
[2]: from tensorflow.keras.datasets import mnist
     from tensorflow.keras.utils import to_categorical

     (train_images, train_labels), (test_images, test_labels) = mnist.load_data()
     train_images = train_images.reshape((60000, 28, 28, 1))
     train_images = train_images.astype('float32') / 255
     test_images = test_images.reshape((10000, 28, 28, 1))
     test_images = test_images.astype('float32') / 255
     train_labels = to_categorical(train_labels)
     test_labels = to_categorical(test_labels)

     model.compile(optimizer='rmsprop',
                   loss='categorical_crossentropy',
                   metrics=['accuracy'])
     model.fit(train_images, train_labels, epochs=5, batch_size=64)

     test_loss, test_acc = model.evaluate(test_images, test_labels)
     test_acc

     Epoch 1/5
     938/938 [==============================] - 12s 13ms/step - loss: 0.4080 - accuracy: 0.8654
     Epoch 2/5
     938/938 [==============================] - 12s 13ms/step - loss: 0.0543 - accuracy: 0.9827
     Epoch 3/5
     938/938 [==============================] - 12s 13ms/step - loss: 0.0350 - accuracy: 0.9893
     Epoch 4/5
     938/938 [==============================] - 12s 13ms/step - loss: 0.0259 - accuracy: 0.9926
     Epoch 5/5
     938/938 [==============================] - 12s 13ms/step - loss: 0.0201 - accuracy: 0.9938
     313/313 [==============================] - 1s 2ms/step - loss: 0.0332 - accuracy: 0.9899
[2]: 0.9898999929428101
```

VSCode

安装 Python 支持



选择虚拟环境并信任 notebook



运行 notebook

CNN.ipynb

[3]

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

model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
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, activation='softmax'))

model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_6 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 3, 3, 64)          36928
_________________________________________________________________
flatten_2 (Flatten)          (None, 576)               0
_________________________________________________________________
dense_4 (Dense)              (None, 64)                36928
_________________________________________________________________
dense_5 (Dense)              (None, 10)                650
=================================================================
Total params: 93,322
Trainable params: 93,322
Non-trainable params: 0
```

延伸阅读

- Ubuntu 机器学习环境 (TensorFlow GPU, JupyterLab, VSCode)
- Mac 机器学习环境 (TensorFlow, JupyterLab, VSCode)
- Win10 机器学习环境 (TensorFlow GPU, JupyterLab, VSCode)
- Apple Silicon Mac M1 原生支持 TensorFlow 2.5 GPU 加速（tensorflow-metal PluggableDevice）

参考

- https://github.com/apple/tensorflow_macos
- https://github.com/apple/tensorflow_macos/issues/153

Written by CatchZeng Follow

AI (Machine Learning) and DevOps enthusiast.

0 条评论 未登录用户

说点什么

支持 Markdown 语法 使用 GitHub 登录 预览

来做第一个留言的人吧！