Convolutional Neural Network

and Other AI Technology with TensorFlow on Robots

Part 1A:
Image Classification Model with
Convolutional Neural Network

Disclaimer: TensorFlow is an open-sourced AI project developed by Google; Python is a programming language developed by the Python Foundation; Java is a programming language developed by Oracle; this work is made by the Absolute Hack Robotics Team at University Lake School which is not funded nor affiliated with any of the organizations aforementioned.

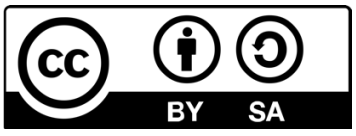

Table of Contents

Chapter 1: Installing TensorFlow

1.1 System Requirements

One of the key features of TensorFlow is that it can be run on multiple platforms so that you may use any computer operating system to install TensorFlow. However, due to the nature of all computer science projects, this documentation is only tested with Debian Linux. The method described may not be compatible with other operating systems.

The operating system able to run TensorFlow includes yet not limited to the following:

- Ubuntu (any Debian Based Linux Operating System) 16.04 or later
- Windows 7 or later (with a C++ redistribution)
- macOS 10.12.6 (Sierra) or later
- Windows Subsystem for Linux on Windows 10 or later versions

In order to install and run TensorFlow, the operating system must be running Python3. From version 3.8 to 3.11, although more than a third of the packages are not yet supported on Python 3.11. Therefore, the best version to run TensorFlow is, according to tests run by the author, Python 3.9.x.

1.1.1 Say Hello to the Little Tool Kit

There are several different tools that will not only be used by TensorFlow but also the programmer (i.e.: you) when training a Convolutional Neural Network (later referred to as CNN). In this section, the process of installing these tools will be described.

The first tool and the one that will be used most commonly is a system package manager. The manager is different for each respective operating system. For all Linux distributions that are based on Debian, which includes Ubuntu, Mate, Mint, and of course, Debian, the manager — apt, or advanced package tool — is integrated into the system. Windows and macOS users have to install their respective package manager manually.

First of all, you have to install Python on the machine on which you will be running TensorFlow.

Method 1: For Windows and macOS Users

- Go to https://www.python.org navigate to the "Downloads" tab in the Ribbon, download the installation media for the appropriate operating systems.
- Click on the download media and follow the on-screen guide to install Python.

Method 2: For Linux Users

- Open a terminal and run the command:

```
sudo apt install python3 -y
```

- Enter root password and wait for it to finish.

With the properly installed Python programming language, it should integrate the module manager for Python exclusively —pip. To use pip, open a terminal session and type in pip.

1.2 Installing TensorFlow

To install TensorFlow, the most preferred method is to install using the pip tool rather than manually compile from the source. The first step before anything is to update the pip tool. It may be updated since the previous use. To do so, run the following command in terminal:

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

> ⓘ Tips
>
> Always update any software before the actual use of it. Especially for package managers since some of the links for packages are updated after the previous use.

After pip is installed and updated, you are ready to install TensorFlow on your machine. At this point, please determine that do you need the aid of a GPU when training the model of CNN. Certain graphics cards do not support GPU optimization when running or training a model. For example, Apple Silicon chips [M1, M2 …] (both CPU and GPU) are heavily optimized for running CNN algorithms, other chips manufactured by other companies may not have the same hardware optimizations. Luckily, TensorFlow has two different distributions for both scenarios.

1.2.1 Installing on Linux Distributions

To install TensorFlow with GPU (in this case, NVIDIA graphics cards) support, first install the NVIDIA Graphics Driver at https://www.nvidia.com/Download/index.aspx. Then in terminal, run the following commands:

```
pip install tensorflow[and-cuda]
```

If you wish to run TensorFlow natively on CPU only, run the following command in the terminal:

```
pip install tensorflow
```

1.2.2 Installing on Windows (native) or Windows Subsystem for Linux

To install TensorFlow on Windows (native) machines, the version must be 7 or higher. The first step is to install Microsoft Visual C++ Redistributive package. Go to the page at https://support.microsoft.com/help/2977003/the-latest-supported-visual-c-downloads and scroll down to the section named Visual Studio 2015, 2017, and 2019. Then download the Microsoft Visual C++ Redistributive package for the appropriate architecture.

Since the lack of support on native Windows machines, it is required to create a virtual environment with the tool named Miniconda. Download the installer at https://repo.anaconda.com/miniconda/Miniconda3-latest-Windows-x86_64.exe and run the software to install Miniconda.

Once it is properly installed, run the following command in Windows Command Prompt:

```
conda create --name tensorflow python=3.9
```

if you wish to enable GPU support in a conda environment, run the following command:

```
conda install -c conda-forge cudatoolkit=11.2 cudnn=8.1.0
```

then you may install TensorFlow with pip as if it in other environments with the command:

```
pip install tensorflow
```

> ⚠ Caution
>
> TensorFlow 2.10 is the last version of TensorFlow that will support GPU optimization when running on local. Since this project uses a later version of it, please install TensorFlow as described in this section for the best result.

1.2.3 Installing on macOS

Since TensorFlow does not support any Apple Silicon GPU acceleration, the user requires it to run TensorFlow on a CPU only, even though the optimization is extremely optimal, run the command as follows:

```
pip install tensorflow
```

ⓘ Tips

Before installing TensorFlow on macOS, please check the version of Python and pip first with commands:

```
Python3 --version
Python3 -m pip --version
```

It is required to have Python 3.9 – 3.11 and pip version >=20.3

⚠ Caution

Before installing TensorFlow on macOS, please check the version of Python and pip first with commands:

```
Python3 --version
Python3 -m pip --version
```

It is required to have Python 3.9 – 3.11 and pip version >= 20.3

1.2.4 Verifying Installation

To verify if TensorFlow has been properly configured and installed, in the terminal, run the following command:

```
python3 -c "import tensorflow; print(tensorflow.__version__)"
```

If the code does not return any error or traceback, then TensorFlow is properly installed.

1.2.5 Quick Trouble Shooting

There are several issues may occur during the process to install TensorFlow. In this section, a table is provided to resolve some simple issues generated by either the misuse of certain commands or errors that occurred randomly.

- Error: No Module Named "tensorflow"

This error is returned due to the fact that TensorFlow is not properly installed with the destinated commands. To verify that TensorFlow is not installed on the current Python environment, use command:

```
pip freeze | grep tensorflow
```

If nothing is returned, then TensorFlow is not properly installed. To resolve this issue, depending on the type of operating system you are using, refer to sections 1.2.1, 1.2.2, or 1.2.3.

- Error: “Could not find a version that satisfies the requirement tensorflow”

If you see this error, it means that pip cannot find the correct version of TensorFlow to install in the current configuration. Consider replacing tensorflow in the command with “tensorflow==2.13.0”. And this should resolve the issue.

1.3 Installing Necessary Python Modules

Within this project specifically, we will be using TensorFlow Lite, or the simpler version of TensorFlow powered by the high-level application programming interface (later referred to as an API): Keras and a module named tflite_model_maker. However, in order to run those modules there are several different packages that need to be installed to Python if not installed already.

The following is a list of modules needed by this project and their respective version information. If version is not specified, then any version can be used (or ignore version specification when installing)

• tensorflow version 2.13	• tflite
• fire	• pandas
• librosa	• lxml
• neural-structured-learning	• numba
• numpy	• scann
• sentencepiece	• tensorflow-addon
• tensorflow-datasets	• tensorflow-model-optimization
• tf-model-official	• tf-model-maker
• tflite-support	• matplotlib
• tensorflow-hub	• tensorflowjs
• urllib3	• gin-config version 0.1.1
• immutabledict	• google-api-python-client
• kaggle	• oauth2client
• opencv-python-headless	• py-cpuinfo
• pycocotools	• sacreblue
• seqeval	• tf-slim

It is highly encouraged to install these modules separately where each line of command will only install one module at a time. This is because some of the modules are not compactable with certain versions of other modules. And pip will by default change the conflicting version of the modules which will create more conflicting modules. If you see pip doing this, abort the operation immediately and add the flag “--no-dependencies” to the end of the pip command.

Chapter 2: Collecting Data and Creating Models With Python

2.1 Gathering Image Data

Data is vital when creating a model using CNN. This algorithm feeds on the images you have provided. Therefore, it is critical to understand what kind of pictures are good for a CNN model and what are not. Furthermore, the classification of such is critical when training the model. Generally, a good dataset covers the following criteria:

- Huge amount of samples
- Each sample contains the object only
- Overall size of the dataset should be relatively small
- Contains a varieties of postures of the object

Typically, a commercial AI model contains at least 1,000 samples per category. In the following sections, you will explore the basic skills to take pictures that fulfill each of the preceding criteria.

☑ Fun Fact

The famous chat AI — ChatGPT is by far the largest text-processing AI human ever generated with more than 175 billion parameters and it started with a 570 GB dataset.

2.1.1 Taking Photos of the Object

For this project, we will be using photos of a specific object as the sample feed to the CNN model created by TensorFlow. It is crucial to bear in mind when taking pictures of the object that it should always contain the object only with minimal background information as possible. The following figures demonstrated a correct and acceptable form of picture:

Figure 1: Correct

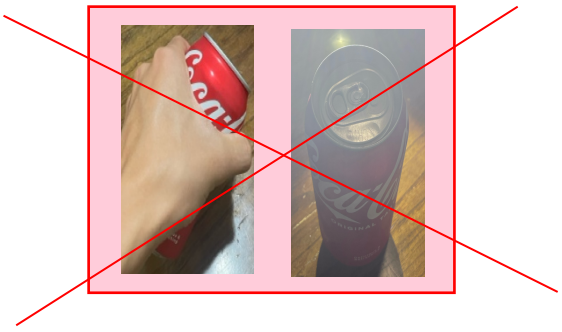

Figure 2: Incorrect

Figure 1 is correct since it only contains the intended object, or the Coke can with minimal backgrounds and was taken from multiple different angles that allowed the algorithm to learn the dimensions and physical characteristics of the Coke can. However, Figure 2 is not acceptable since in the picture on the left, the Coke can is covered by a hand holding it; this will

disturb the algorithm of CNN that prohibits the further characterization of the object. On the right side, the lighting of the picture blurs the object which will decrease the accuracy of the color identification part of the algorithm.

Therefore, the following is a list of criteria that you should bear in mind when taking photos of the object:

- Place the object in a bright lighting conditions with minimal backgrounds
- Adjust camera angles so that the object is in the center of the view
- Take the pictures from multiple angles
- Do not block any parts of the object

2.1.2 Creating and Understanding the File Structure

Once the pictures or the raw data has been collected, they should be immediately renamed and organized if not already do so. To organize the files, you first have to understand and create a project folder on your designated machine.

On your machine, create a folder wherever is appropriate and rename it to "tensorflow_datasets"; this will be your project folder where other files and scripts created are stored. Under this folder, create a subfolder named "data". Within the "data" folder, create several different subfolder which each categorize a class of images.

When these folders are created, then classify all images into those folders. The following diagram demonstrated a correctly labeled and organized project folder:

```
.
└── tensorflow_dataset/
    └── data/
        ├── maples/
        │   ├── m1.jpg
        │   ├── m2.jpg
        │   └── ...
        ├── pines/
        │   ├── p1.jpg
        │   ├── p2.jpg
        │   └── ...
        ├── walnut/
        │   ├── w1.jpg
        │   ├── w2.jpg
        │   └── ...
        └── coconut/
            ├── c1.jpg
            ├── c2.jpg
```

2.1.3 Double Check for File Type

There are many types of files that TensorFlow cannot handle, especially for this project, you would like to avoid files such as HEIC, WBPG, or CR2. The following is a list of acceptable materials that TensorFlow CNN can use when training and deploying.

- JPG
- JPEG
- PNG
- GIF

For the most common use, consider adding all pictures in the form of .jpg and .jpeg. If your file is not in the correct format, use converting tools to change the file type to the aforementioned ones.

> ⚠ Caution
>
> Be aware that directly changing the file extension name in order to change the file type is not a reliable method. Although for the most time, it will be fine; however, in certain circumstances, this will break the file and make it not accessible.

2.2 Generate the CNN Model with Python Locally

So far, we have successfully installed and gathered data for the CNN model. Now it is time to generate this model. In this section, we will be generating a CNN model with Python scripts. Do note that the model will be generated locally with your hardware. This process is extremely time consuming and power consuming. Do consider generating the model online with Google's digital platform (see section 2.3) if you suspect your computer cannot perform this algorithm.

2.2.1 Writing the Python Script for Generation

To generate a model, you have to write a Python script that will be executed to train the model. Notice that in order to simplify the procedure, we used the "tflite-model-maker" module of Python; therefore, please check that this model is properly installed.

Open the IDE (Integrated Development Environment) of your choice and open the project folder. Create a script file named "init.py" and first import the necessary modules;

```
import os

import numpy as np
import tensorflow as tf
assert tf.__version__.startswith('2')

from tflite_model_maker import model_spec
from tflite_model_maker import image_classifier
from tflite_model_maker.config import ExportFormat
from tflite_model_maker.config import QuantizationConfig
from tflite_model_maker.image_classifier import DataLoader
import matplotlib.pyplot as plt
import pathlib
```

Once this code has been written, append the following code to the script file:

```
img_pth = pathlib.Path('images').absolute()
data = DataLoader.from_folder(img_pth)
train_data, test_data = data.split(0.9)

model = image_classifier.create(train_data, batch_size=1)

loss, accuracy = model.evaluate(test_data)

model.export(export_dir= ".")
```

Now the model is ready to be trained. Go to your terminal and run the following command:

```
python3 init.py
```

if no errors or trackbacks are reported, congratulations, the model is in its training process. This process typically takes up to hundreds of hours to train models regarding the hardware of your computer. If you suspect that your computer is not capable of performing such, see section 2.3 to train the model using a digital platform. One hardware example that this guide is tested on is the following: **to do: add hardware config.** .

An additional tip to train the model is that seek for processors that have built-in AI training accelerations and optimizations such as the Apple Silicon chips are famous for their AI optimizations as well as other AMD chips.

2.2.2 Quick Trouble Shooting

The most common occur during the training process of a model are possibly missing modules and packages. To fix this error, consider using pip to install the module manually. This is possibly caused by the "--no-dependencies" flag which you may specify when installing other modules.

- TensorFlow is not detecting the GPU (and / or cannot use hardware acceleration)

This can be caused by multiple reasons such as your computer is using an outdated driver which TensorFlow does not support any more or missing essential dependencies which are necessary for the acceleration. To resolve this, try one of the following options:

1) Update your driver for GPU
2) If you are working on WSL, install CUDA
3) Just use the CPU, I guess

- Package version conflict (either due to pip or other package manager)

If this error is returned by pip, carefully read the error message which is typically in red and adjust the version of such package(s) with the following command accordingly: (use the –no-dependencies flag as needed)

```
pip install [package name]==[version] --update
```

Try not to change the version to the latest, since when this guide is written and the version of TensorFlow we are using, many of the packages are not compatible with others which are in later versions. The one trick that will possibly solve the majority of the version issue is to use the oldest version compatible to other packages.

- Error: Image size is zero

This error is caused by the specified directory does containing any training photos or the pictures are not in the required file structure. To resolve this issue, check the specified folder and the spelling in the code; you may misspell the word or added unwanted space or characters to the path. Or refer to 2.1.2 to double check the file structure.

- Error: File type not supported

This error can be triggered by two possible issues: either, quite literally, the file type is not supported by TensorFlow. (to check the acceptable file type, see section 2.1.3) or that the file has a bad encoding method which TensorFlow does not appreciate. To resolve this issue, check if the file is broken or switch to a different type of file as suggested.

2.3 Generate the CNN Model with Python on Google Colaboratory

If you suspect that your computer cannot handle the calculations locally, consider training the model on the Google Colaboratory which is a web-based platform that allows you to train the model with minimal hardware setup. Several advantages of this platform are the platform integrated with the necessary setup for all the development needs.

2.3.1 Stranger in the Strange Platform

To start, go to https://colab.research.google.com and create a new notebook. Then copy all the code in the previous section to the notebook that you created. Except that change the variable "img_pth" to the following:

```
img_pth = os.path.join(os.path.dirname(image_path), 'images')
```

On the left-hand panel, there is a button that allows you to upload pictures as it is shown in Figure 3

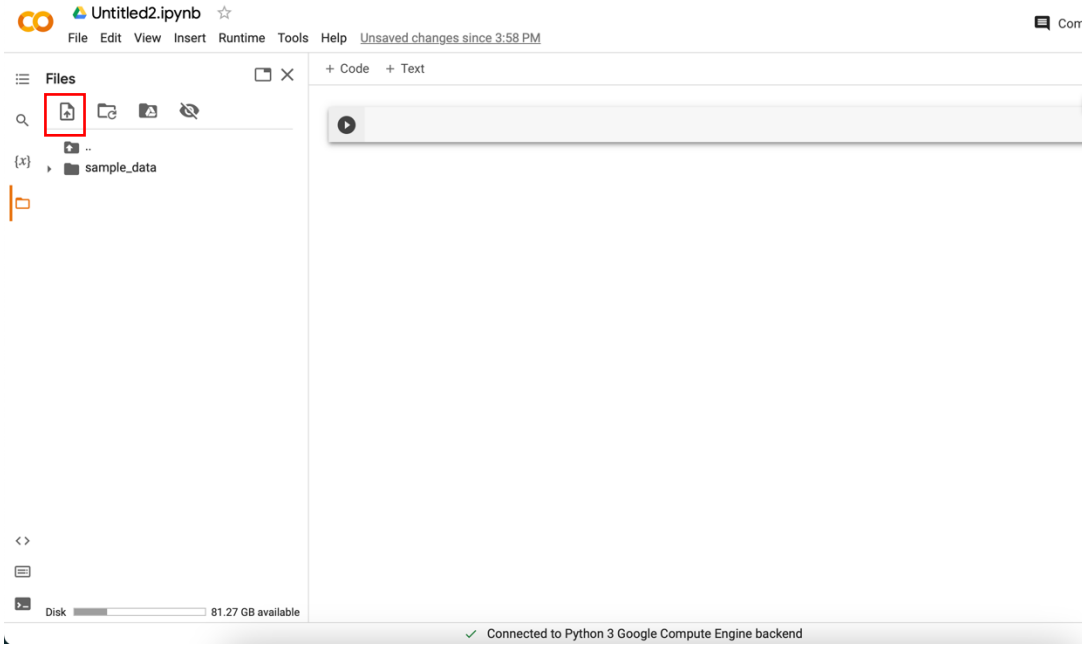

Figure 3: uploading data

Ensure that the file structure is the same as it is described in 2.1.2. When you finish, the final notebook should be similar to the following in Figure 4.

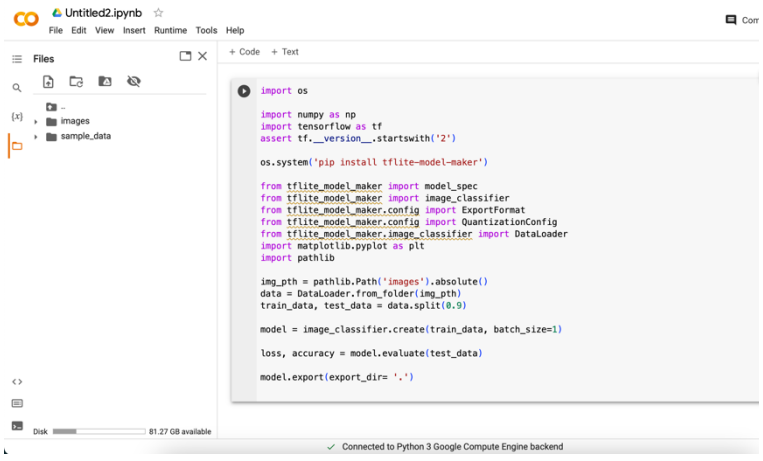

Figure 4: finalized notebook

when you finished finalizing the notebook, click on the run button on the left corner of the code snip and this will allow the code to be executed on Google's virtual runtime platform instead of your computer.

When the training process is finished, right-click on the model file named "model.tflite" and select download the model.

Congratulations, you have successfully trained a model with TensorFlow Lite Model Maker.

> ⚠ Caution
>
> As when this guide is written, in order to enable the runtime model for Google Colab, a subscription to Google Colab Pro is required.

Chapter 3: Interacting with the TensorFlow Lite Model

Now that we have obtained the model we trained, it is time to apply this to actual production. In this section, the method by which one can produce their own application with TensorFlow. This process consists of the following general steps:

- Loading the model from the .tflite file
- Using the Keras API provided by the TensorFlow Lite SDK to access the file
- Optimize as needed
-

3.1 General

Appendix A: TensorFlow Image Classification Model QRH

This Quick Reference Handbook (QRH) is designed so that in certain circumstances, you can find the solution quickly, either within this set of documents or on the Internet.

Directory of QRH

A.1 Hardware Configuration

Within this section, you will find resources related to issues that happen in hardware configurations and issues which may arise when inappropriate configuration is used.

A.1.1 Operating System Not Listed in Section 1.1

Within Section 1.1, the operating system is the one for which the guide has been tested and appears to be functioning. For now, the only platform that TensorFlow for sure does not run on is ChromeOS.

A.1.2 Lack of GPU Support (Acceleration)

This issue can be considered common when speaking with a training model. Check the following checklist to see if you need GPU acceleration:

Does (Is) your computer ….

1. Have a powerful CPU that has more than 5 cores? (y/n)
2. Have a graphical user interface? (y/n)
3. Running a power-consuming operating system (such as Windows >= 8) (y/n)
4. Running an operating system that does not allow terminal inputs, such as ChromeOS (y/n)
5. A laptop that does not have a chip that supports AI acceleration? (y/n)
6. Have >= 8GB of RAM (Random Access Memory, not storage space) (y/n)
7. Generally open Google Chrome within 5 seconds? (y/n)

If you answered more than (including) 2 ns in the checklist above, you should consider using an online platform as it is described in Section 2.3.

If you answered less than 2 ns, you may want to enable hardware acceleration for best performance when dealing with large numbers of samples, although it is not required.

A.2 Software Installation Issues

A.2.1 Command Not Found