cfav-dev fix: add protobuf to requirements.txt 6 months ago

Name	Name	Last commit date
..		
assets	repo - add top level dir for e...	3 years ago
images	repo - add top level dir for e...	3 years ago
README.md	C1W1 lab - Add note to pre...	last year
client.ipynb	C1 W1 ungraded lab 1: adde...	3 years ago
mac_m1.md	C1W1 lab - Add additional n...	last year
requirements.txt	fix: add protobuf to require...	6 months ago
server.ipynb	repo - add top level dir for e...	3 years ago

README.md

Ungraded Lab - Deploying a Deep Learning model

Introduction

Welcome to the first week of Machine Learning Engineering for Production Course 1. During this ungraded lab you will go through the process of deploying an already trained Deep Learning model. To do so, we will take advantage of the user-friendly library fastAPI that provides a nice REST API framework.

This tutorial is specifically designed to run locally on your machine. This can be done via 2 methods: using `Python Virtual Environments` or using `Docker`.

Both approaches should yield the same result. If you already have a conda installation available on your computer, we recommend that you use the virtual environment method. If this is not the case, choose the Docker method as it is easier to set up.

As a general note, the commands in this tutorial are meant to be run within a terminal. To begin you need to **clone this repo in your local filesystem and `cd` to the week1-ungraded-lab directory**.

To clone the repo use this command:

```
git clone https://github.com/https-deeplearning-ai/machine-learning-engineerin
```

or for cloning via SSH use:

```
git clone git@github.com:https-deeplearning-ai/machine-learning-engineering-fo
```

If you are unsure which method to use for cloning, use the first one.

The `cd` command allows you to change directories. Assuming you are at the directory where you issued the cloning command, type the following on your terminal.

```
cd machine-learning-engineering-for-production-public/course1/week1-ungraded-l
```

This will bring you to the `week1-ungraded-lab` directory. The `ls` command allows you to list the files and directories. Type `ls` and let's take a quick look at the content inside `week1-ungraded-lab` directory:

```
.
└── week1-ungraded-lab (this directory)
    ├── images (includes some images from ImageNet)
    ├── server.ipynb (Part 1 of the ungraded lab)
    ├── client.ipynb (Part 2 of the ungraded lab)
    └── requirements.txt (python dependencies)
```

Method 1: Python Virtual Environment with Conda

Prerequisites: Have conda installed on your local machine.

You will use Conda as an environment management system so that all the dependencies you need for this ungraded lab are stored in an isolated environment.

Conda includes a lot of libraries so if you are only installing it to complete this lab , we suggest using miniconda, which is a minimal version of conda.

1. Creating a virtual Environment

Now we assume that you either successfully installed conda or that it was previously available in your system. The first step is creating a new developing environment. Let's set a new environment with python 3.8 with this command:

```
conda create --name mlep-w1-lab python=3.8
```

After successfully creating the environment, you need to activate it by issuing this command:

```
conda activate mlep-w1-lab
```

At this point, you will do all your libraries installation and work in this environment. So, whenever working on this ungraded lab, check the mlep-w1-lab environment is active.

Note: If you have a Mac M1 then read this guide before doing the next steps or try the version of this lab that is hosted in Coursera.

2. Installing dependencies using PIP

Before proceeding, double check that you are currently on the `week1-ungraded-lab` directory, which includes the `requirements.txt` file. This file lists all the required dependencies and their respective versions.

If you are on a Mac M1 be sure to have installed Tensorflow and OpenCV as described in the guide and deleted these libraries from the `requirements.txt` file.

Now use the following command to install the required dependencies:

```
pip install -r requirements.txt
```

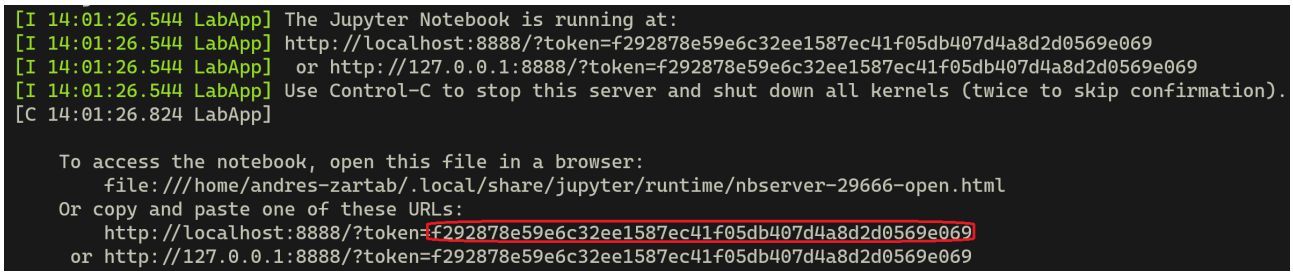
This command can take a while to run depending on the speed of your internet connection. Once this step completes you should be ready to spin up jupyter lab and begin working on the ungraded lab.

3. Launching Jupyter Lab

Jupyter lab was installed during the previous step so you can launch it with this command:

```
jupyter lab
```

After execution, you will see some information printed on the terminal. Usually you will need to authenticate to use Jupyter lab. For this, copy the token that appears on your terminal, head over to http://localhost:8888/lab and paste it there. Your terminal's output should look very similar to the next image, in which the token has been highlighted for reference:



4. Running the notebook

Within Jupyter lab you should be in the same directory where you used the `jupyter lab` command.

Look for the `server.ipynb` file and open it to begin the ungraded lab.

To stop jupyter lab once you are done with the lab just press `Ctrl + C` twice.

And... that's it! Have fun deploying a Deep Learning model! :)

Method 2: Docker

Prerequisites: Have Docker installed on your local machine.

Docker is a tool that allows you to ship your software along with all the dependencies that it needs to run. You can download the free version here.

Installing Docker is straightforward for Linux and Mac users but might be more challenging for Windows users. If you use Windows we recommend you stick with the first method to run this ungraded lab. However if you want to go through with this method you will need to install Docker Desktop and WSL2 (Windows Subsystem for Linux 2). By doing this you will enable a real Linux kernel inside your Windows OS and will get full usage out of Docker. For a better experience we also recommend to install the Windows Terminal. This is a great setup to do machine learning related work on Windows but it requires a lot more setup.

Note: If you are on a M1 Mac use the `Creating a virtual Environment` method or try the version of this lab that is hosted in Coursera.

1. Pulling the image from Docker hub

Images are an important concept within the Docker ecosystem. You can think of them as the compilation of all the elements (libraries, files, etc) needed for your software to run.

Using the following command will download or pull the image necessary to run this ungraded lab locally:

```
docker pull deeplearningai/mlepc1w1-ugl:jupyternb
```

2. Running a container out of the image:

Images can also be thought of as the blueprints for containers, which are the actual instances of the software running. To run a container using the image you just pulled, double check that you are currently on the `week1-ungraded-lab` directory and use this command:

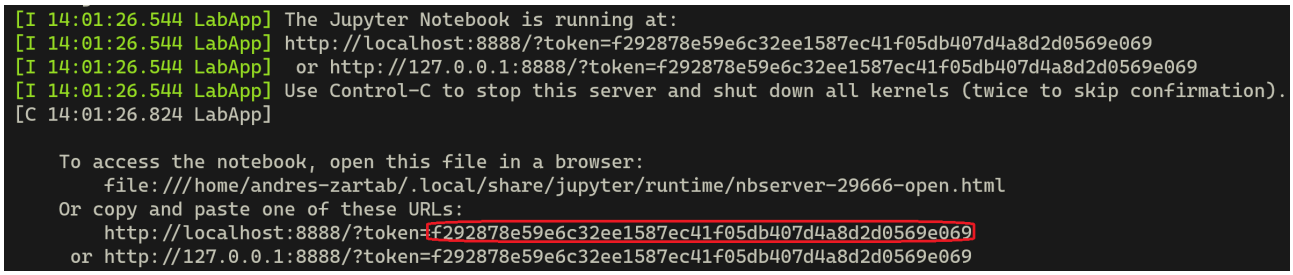
```
docker run -it --rm -p 8888:8888 -p 8000:8000 --mount type=bind,source="$(pwd)
```

Let's break down this command and its flags:

- -it: Runs the container in an interactive mode and attaches a pseudo-terminal to it so you can check what is being printed in the standard streams of the container. This is very important since you will have to **copy and paste the access token for Jupyter lab**.
- --rm: Deletes the container after stopping it.
- -p: Allows you to map a port in your computer to a port in the container. In this case we need a port for the Jupyter server and another for the server you will run within the ungraded lab.
- --mount: Allows you to mount a directory in your local filesystem within the container. This is very important because if no mounts are present, changes to files will not persist after the container is deleted. In this case we are mounting the current directory `week1-ungraded-lab` onto the `/home/jovyan/work` directory inside the container.

When the container starts running you will see some information being printed in the terminal. Usually you will need to authenticate to use Jupyter lab, for this copy the token that appears on your terminal, head over to http://localhost:8888/lab and paste it there.

Your terminal's output should look very similar to the next image, in which the token has been highlighted for reference:



Once you have authenticated, click in the `/work` directory and you should see all of the files from your current local directory. Look for the `server.ipynb` file and open it to begin the ungraded lab.

To stop the container once you are done with the lab just press `Ctrl + C` twice. This will also delete the container.

And... that's it! Have fun deploying a Deep Learning model! :)