Docker Container Documentation

Team 24 : Jiye Choi, Sora Ryu

This program uses a container to hold the framework that is PyTorch. The application users are allowed to send the image that is request, and when the container is on, it will take the request and return the result back to the user.

The server which handles HTTP requests is included in a docker container. The container uses Python3.7 image as the launching point. Then, it installs the required dependencies which are described in requirements.txt file. It installs Flask, torch, and torchvision. After copying all the files located in the current directory into the image, expose the port 5000 from container. Finally, start the python-flask application and make the application externally visible from outside of the container, by specifying '--host=0.0.0.0'.

How to run the program in a docker container

From a docker file which contains all the commands that is required for creating an image, docker image and container are built one by one. The docker file is already created, and please follow the steps in order to run the classifier.

1. Install Docker

- From this link, please install docker:
  https://www.docker.com/products/docker-desktop
  If the docker is already installed, you can skip this step.

2-1. Build Docker Image manually

- Build the docker image from Docker file by running this command (root directory):
  ```
  docker build -t pytorch-docker .
  ```
- To check if the image has successfully built, please run this command:
  ```
  docker images
  ```
  You can check that docker image has built with the name *pytorch-docker*.

2-2. Use the pre-built Docker Image

- If you want to skip manual docker image building process (2-1), please download the docker image in the root directory from this link:
  https://drive.google.com/file/d/10Z51z_KwiPyjVC1mf5b-9clxJ7668ULW/view?usp=sharing

- Then, load docker image by running this command:
  `docker load < pytorch-docker.tar.gz`
  For Windows: `docker load -i pytorch-docker.tar.gz`
- To check whether the image has successfully loaded, please run this command:
  `docker images`

3. Run the image as a container

- `docker run -d -p 5000:5000 --name http-server pytorch-docker`
  Docker runs in a detached mode (running in a background) if we use `--detach` or `-d` for short. Expose port 5000 inside the container to port 5000 outside the container, using the `--publish flag` on the `docker run` command. Name a container with *http-server*, by passing the `--name flag` to the `docker run` command.
- To check if the container is running, please run this command:
  `docker ps`
- In order to make sure that the server is running properly, please run this command:
  `curl localhost:5000`If the output is *Server is Running!*, it's running properly.

4. Query the server using images

- Please make sure that images are located in the root folder.
- Send the POST request with the test image file to the http server, by running the following `curl` command:
  `curl -F "file=@[your test image file]" http://127.0.0.1:5000/upload`
  (e.g, `curl -F "file=@chicken.jpeg" http://127.0.0.1:5000/upload`)
- Then, the server will take the image as an input, implement a simple image classification using a pre-trained Densenet-121 model for Pytorch, and return a classification as an output.
  (e.g, `Imagenet_id : n01514668 , Classification : cock`)
- After the testing is finished, please run this command and stop the container running:
  `docker stop http-server`

5. Test script

- To test this server with test images provided in this folder, please run test.sh script.
- Before you run this test script, please make sure Docker is running on your computer.
- Then, run this command in the root folder.
  `bash test.sh`