Docker Container Documentation & Testing

Docker-compose.yml and Dockerfile

A *docker-compose* file is used to build a docker container to run the inference server. The *docker-compose.yml* file is used to define all the *services* required to run an application in a configuration file and is beneficial to spin up containers for multiple/all of the services at once. A service definition contains configurations that are applied to each container started for that service. The *'version: 3.7'* specifies the format of these definitions. The *'build'* property is used to specify to docker what to compose. It can be a string containing a path to the build context. This directory has the *Dockerfile* used to set up the environment for the container. The keyword *'image'* is used to specify a name for the built image. The keyword *'volumes'* is used to specify a list of mappings between the directory with the code in the local system to the working directory in the container. Similarly, *'ports'* is used to specify a list of mappings between the port on the local system to one used by the server running on the container.

In the *Dockerfile* the base image is specified using the *'FROM'* keyword. The contents of the current directory are copied into the working directory of the container using the *'WORKDIR'* and *'COPY'* keywords. The Python libraries required to run the server are installed using *'RUN pip install*'. Finally *'ENTRYPOINT'* is used to specify what needs to be executed once the docker container is started from the image.

Exposing port 5000

The way this ML-inference project has been designed, the server runs on the docker container and the client sends a *GET/POST* request from the local system. The *'EXPOSE 5000'* in *Dockerfile* tells docker that the container's services can be listened to via port *5000*. While running the docker container the publish flag *'-p'* exposes the port to the local system. So sending a request to *http//localhost:5000* will send the request to the server running on the docker container.

Execution/Testing

Step by step Execution

1. Run the below commands to build the project container and run the application:
    a. *sudo docker-compose build*
    b. *docker run -p 5000:5000 ml_inference_team_20*
2. Use an existing image from the *images* folder or place a new image in the *images* folder and run command *python send_request.py <image filename>* to run inference.

Automated Testing

For an automated test run that builds the container, runs the application and runs inference on a sample image *shopping-street.jpg*, run *./run_inference.sh*

- *run_inference.sh*: Runs *build_and_run_docker_image.sh* and *send_request.sh* parallelly
- *build_and_run_docker_image.sh:* Builds the docker container and runs the application
- *send_request.sh:* Waits till the application server is active and runs inference on sample image *shopping-street.jpg* using *send_request.py*