



# Running a CNN on NVIDIA Jetson Orin Nano

CS 4391 Introduction Computer Vision

Instructor: Yu Xiang

The University of Texas at Dallas

# JetPack 6.2 image

- <https://developer.nvidia.com/embedded/jetpack-sdk-62>
- JetPack 6.2 is the latest production release of JetPack 6. This release includes Jetson Linux 36.4.3, featuring the Linux Kernel 5.15 and an Ubuntu 22.04-based root file system. The Jetson AI stack packaged with JetPack 6.2 includes CUDA 12.6, TensorRT 10.3, cuDNN 9.3, VPI 3.2, DLA 3.1, and DLFW 24.0.

# Docker

- An open platform that enables you to separate your applications from your infrastructure
- Container
  - A lightweight environment that contains everything to run an application
  - A container is a runnable instance of an image
- Image
  - A read-only template with instructions for creating a docker container

# Jetson-Containers

- An open-source project by NVIDIA engineer Dusty Franklin that provides **prebuilt Docker containers optimized for NVIDIA Jetson devices** (Orin Nano, Xavier, etc.).
- Matching **PyTorch / CUDA / cuDNN / TensorRT / JetPack / L4T** versions correctly.
- <https://github.com/dusty-nv/jetson-containers>

# Step 1: Set up jetson-containers on Jetson Orin Nano

- Follow the system setup in jetson-containers  
<https://github.com/dusty-nv/jetson-containers/blob/master/docs/setup.md>

## Clone the Repo

This will download and install the jetson-containers utilities:

```
git clone https://github.com/dusty-nv/jetson-containers
cd jetson-containers
bash install.sh #run with sudo if needed
```

## Docker Default Runtime

If you're going to be building containers, you need to set Docker's `default-runtime` to `nvidia`, so that the NVCC compiler and GPU are available during `docker build` operations. Add `"default-runtime": "nvidia"` to your `/etc/docker/daemon.json` configuration file before attempting to build the containers:

```
{
  "runtimes": {
    "nvidia": {
      "path": "nvidia-container-runtime",
      "runtimeArgs": []
    }
  },
  "default-runtime": "nvidia"
}
```

You can use vim or gedit to edit this file

Then restart the Docker service, or reboot your system before proceeding:

```
$ sudo systemctl restart docker
```

# Step 2: Set up l4t-pytorch container on Jetson Orin Nano

- Instruction for running containers <https://github.com/dusty-nv/jetson-containers/blob/master/docs/run.md>
- <https://github.com/dusty-nv/jetson-containers/tree/master/packages/ml/l4t/l4t-pytorch>

```
# automatically pull or build a compatible container image
jetson-containers run $(autotag l4t-pytorch)
```

To mount your own directories into the container, use the `-v` or `--volume` flags:

```
jetson-containers run -v /path/on/host:/path/in/container $(autotag l4t-pytorch)
```

# Step 3: Get the new code for running the CNN

- git clone [https://github.com/yuxng/cs4391\\_spring26.git](https://github.com/yuxng/cs4391_spring26.git)
- Or git pull in the repo if you already have it

# Step 4: Test your camera first

- `python test_usb_camera.py`



# Step 5: Copy your checkpoint to Jetson

- You can use box

## Step 6: Run the network

- `python3 infer_cam_or_images.py --split split.json --ckpt run2/part2/pretrained_ft/bes.pt --source cam --cam_id 0`

# Questions?