Edge Device를 활용한 AI 실습

실시간 AI 어플리케이션의 구현 및 응용 탐구

김 종 찬
고신뢰도모델가속기술연구단
국가슈퍼컴퓨팅본부
2025-01-21

KiSTi
www.kisti.re.kr
Korea Institute of
Science and Technology Information

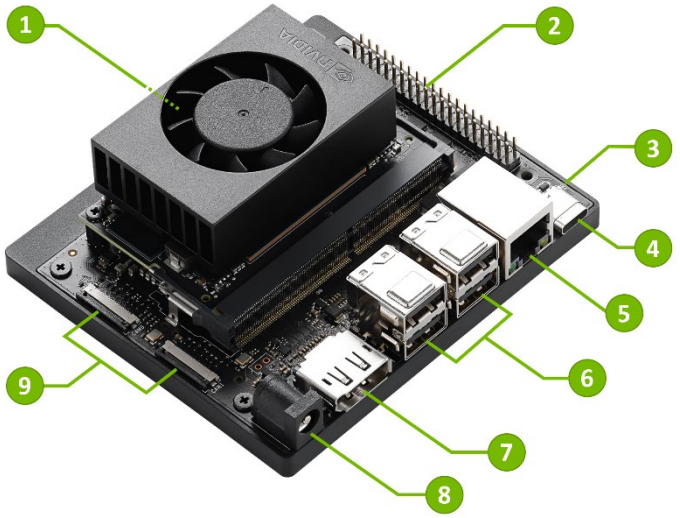
KiSTi 한국과학기술정보연구원
www.kisti.re.kr
Korea Institute of Science and Technology Information

목 차
CONTENTS

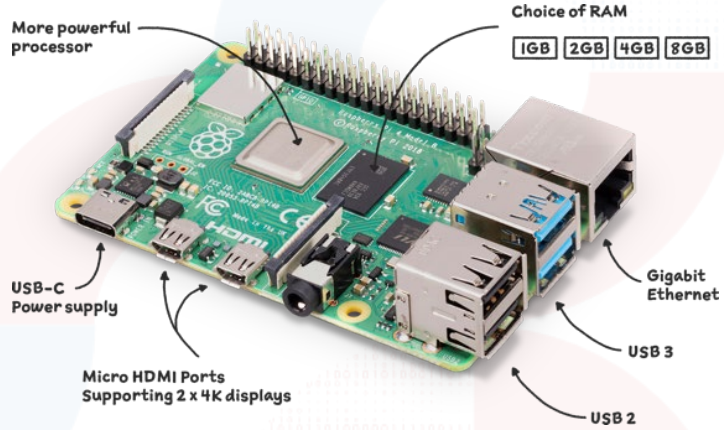
1. Edge Computing 및 AI 개요

* Image made with Dall-E

- Edge Computing의 정의와 필요성
  - "데이터를 소스 가까이에서 처리하는 컴퓨팅 방식"
  - 필요성: 실시간 처리, 낮은 대기시간, 데이터 프라이버시 강화
- Edge AI의 개념과 응용사례
  - 스마트 공장, 의료, 자율주행, IoT
- Edge Computing Device
  - 카메라, 센서 등
  - SBC(Single Board Computers)
    - NVIDIA Jetson, Raspberry Pi
  - 임베디드 시스템
    - Arduino, BeagleBone
  - 산업용
    - HPE Edgeline, Dell EMC Edge Gateway



Jetson Orin Nano (nvidia.com)



Raspberry Pi 4 (raspberrypi.com)

2. 실습 목표

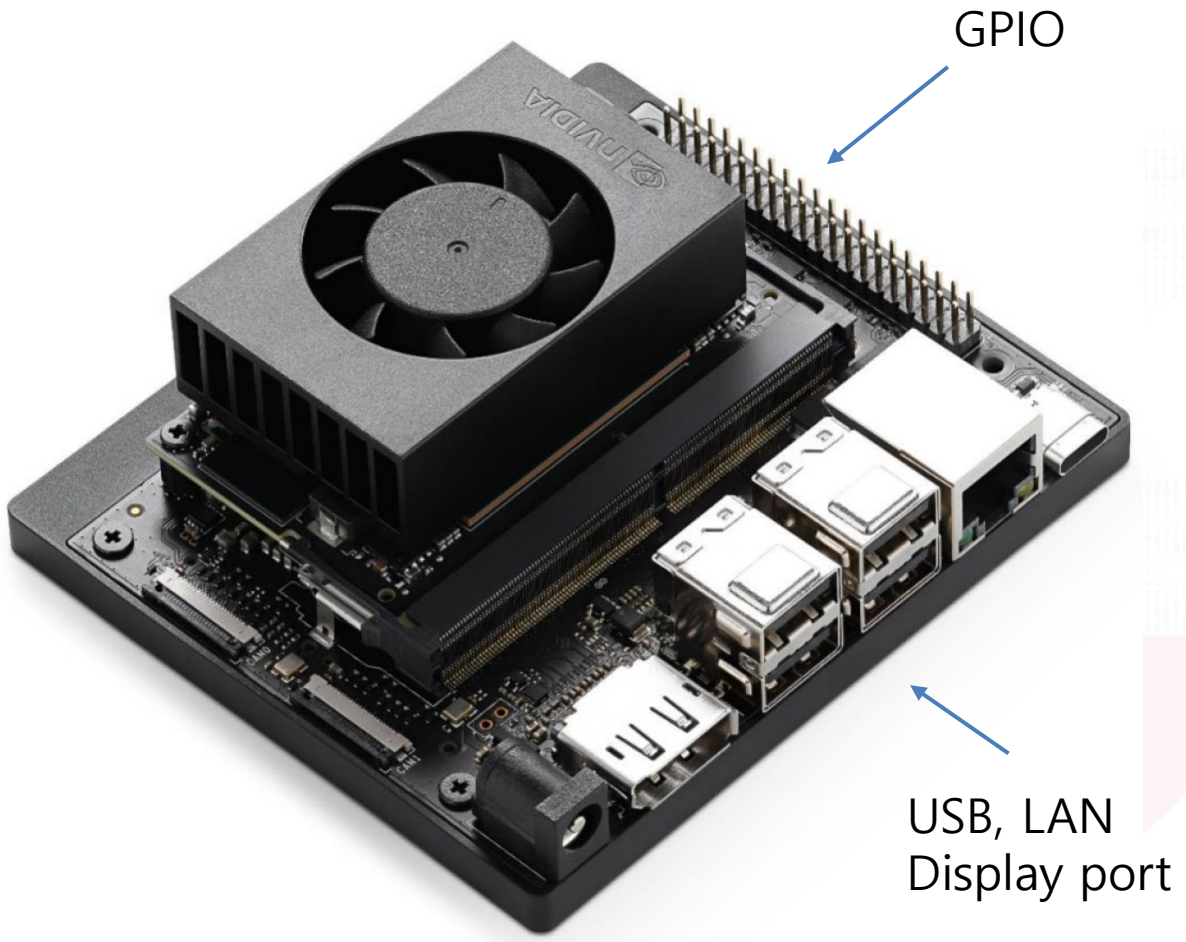
- Jetson Nano Orin에서 AI 모델 실행
  - 목표: 실시간 classification 구현

3. Jetson Orin Nano 소개

- 특징
  - NVIDIA GPU 기반 CUDA 코어
  - 고성능, 높은 에너지 효율
  - TensorRT와 PyTorch 지원

- 영상분석, 로봇 공학, AIoT 프로젝트에 활용

- 성능
  - CPU: 6-core Arm Cortex-A78AE v8.2 64bit
  - GPU: NVIDIA Ampere architecture with 1024 NVIDIA CUDA cores & 32 Tensor cores
  - Memory: 8GB 128bit LPDDR5 (68GB/s)
  - Storage: external via MicroSD slop, NVMe M.2



Jetson Orin Nano (nvidia.com)

4. Jetson Nano Orin 및 실습 환경 설정

- JetPack 설치
  - Ubuntu linux 기반 OS
  - NVIDIA SDK Manager를 통한 JetPack 설치
- 개발 도구
  - Python, PyTorch
  - Jupyter Notebook
  - Docker Image 활용
- 실습
  - 기본 AI 예제 실행
- 과제
  - 실습 예제 응용 문제

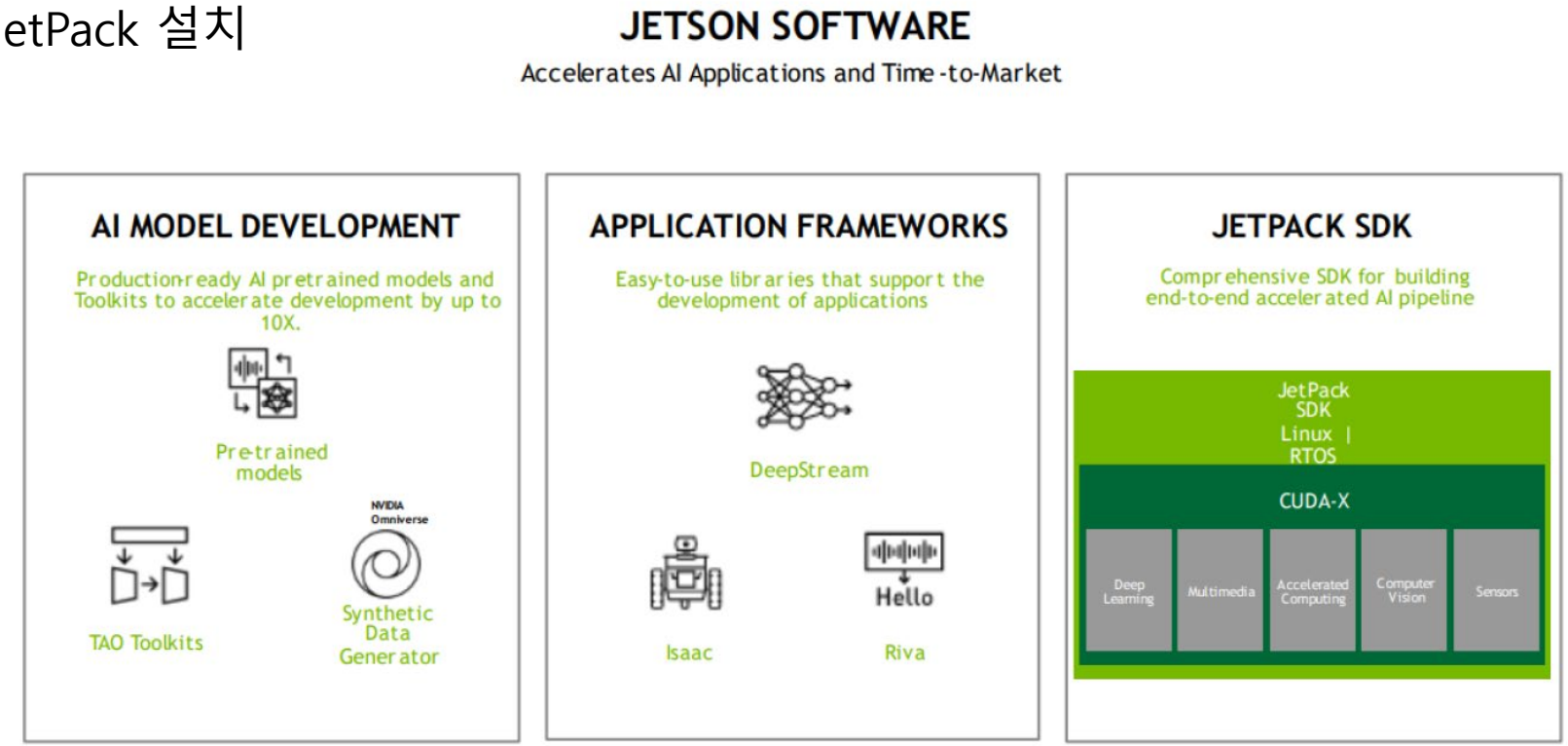
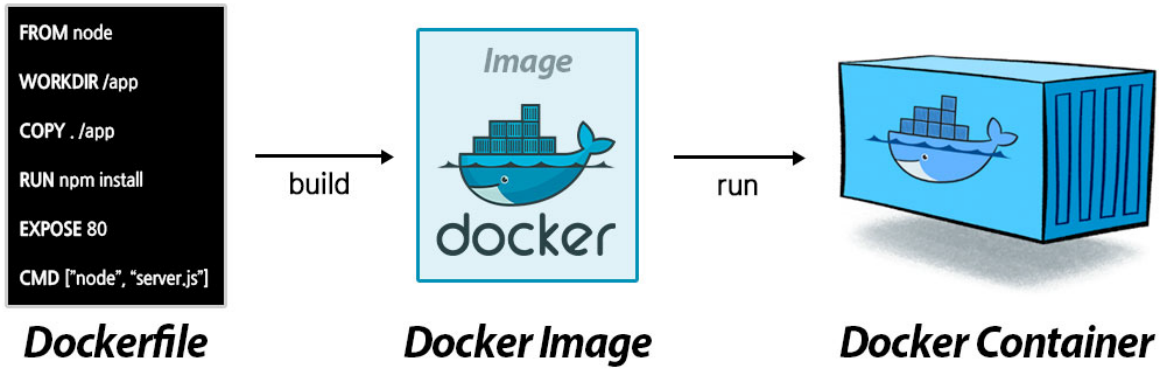


Figure from nvidia.com

5. Docker 란?

- **애플리케이션을 빠르고 쉽게 실행할 수 있도록 도와주는 컨테이너 플랫폼**
- **도커(Docker):** 컨테이너를 생성하고 관리하기 위한 도구
- **컨테이너(Container):** 프로그램, 필요한 파일, 환경 설정 등을 하나로 묶은 작은 박스와 같다.
- 각기 다른 하드웨어에서도 동일한 환경 제공
- 가상머신보다 빠르고 적은 리소스를 이용, 배포와 테스트가 쉬움



- 핵심 개념
  - 이미지(image): 실행 가능한 파일들의 모음
  - 컨테이너(Container): 이미지를 실행한 상태, 프로그램이 실행되는 박스
  - 도커허브(Docker Hub): 이미지들을 공유할 수 있는 공간

Ref) https://docs.docker.com/

5.1 Jetson Nano 연결 및 Docker 환경 설정

1. Jetson Nano USB-c --- Computer 연결 또는 LAN 환경 이용
2. 윈도우 환경에서 터미널 실행: Windows PowerShell 이용
3. Jetson 연결 → `$ssh jetson@192.168.55.1`
   `… password: jetson`
4. 실습파일 다운 → `$git clone https://github.com/jckim1201/25_youth_winter.git`
   `…`
   `Resolving deltas: 100% (21/21), done.`
5. 카메라 연결 확인 → `$ls /dev/video*`

5.1 Jetson Nano 연결 및 Docker 환경 설정

6. Docker 이미지 실행 파일 작성

7. Docker 실행

```
$ls /dev/video*
$mkdir ai-data
$echo sudo docker run --runtime nvidia \
    -it --rm --network host \
    --volume ~/25_youth_winter:/nvdli-nano/data \
    --volume /tmp/argus_socket:/tmp/argus_socket \
    --device /dev/video0 \
    nvcr.io/nvidia/dli/dli-nano-ai:v2.0.3-r36.3.0kr \
    >> docker_run.sh
$chmod +x docker_run.sh
$./docker_run.sh
```

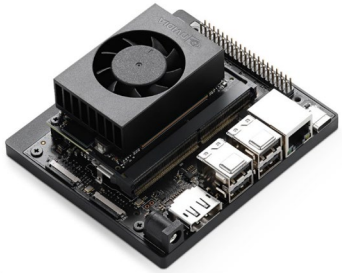
5.2 도커 개발 환경 연결 설명

- Docker 이미지 연결 모습

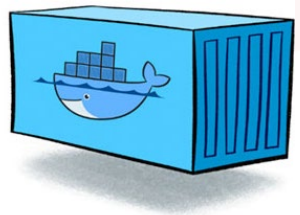
```
Allow 10 sec for JupyterLab to start @
http://***.***.***.***:8888 (password dlinano)
JupterLab logging location: /var/log/jupyter.log
(inside the container)

root@jetson-desktop$
```

- Jupiter Lab 실행
  - 웹브라우저에서 jupyter lab 접속 주소 입력
  - http://***.***.***.***:8888 -> 화면확인!
  - 비밀번호 dlinano

- Docker 이미지 내 파일과 실제 파일의 위치
  - Warning!! : Docker 실행 시 생성된 데이터는 docker 실행이 종료된 이후, 모두 지워 짐
  - 데이터를 실제 저장공간에 남기기 위해 실행 명령어에 저장 위치 설정



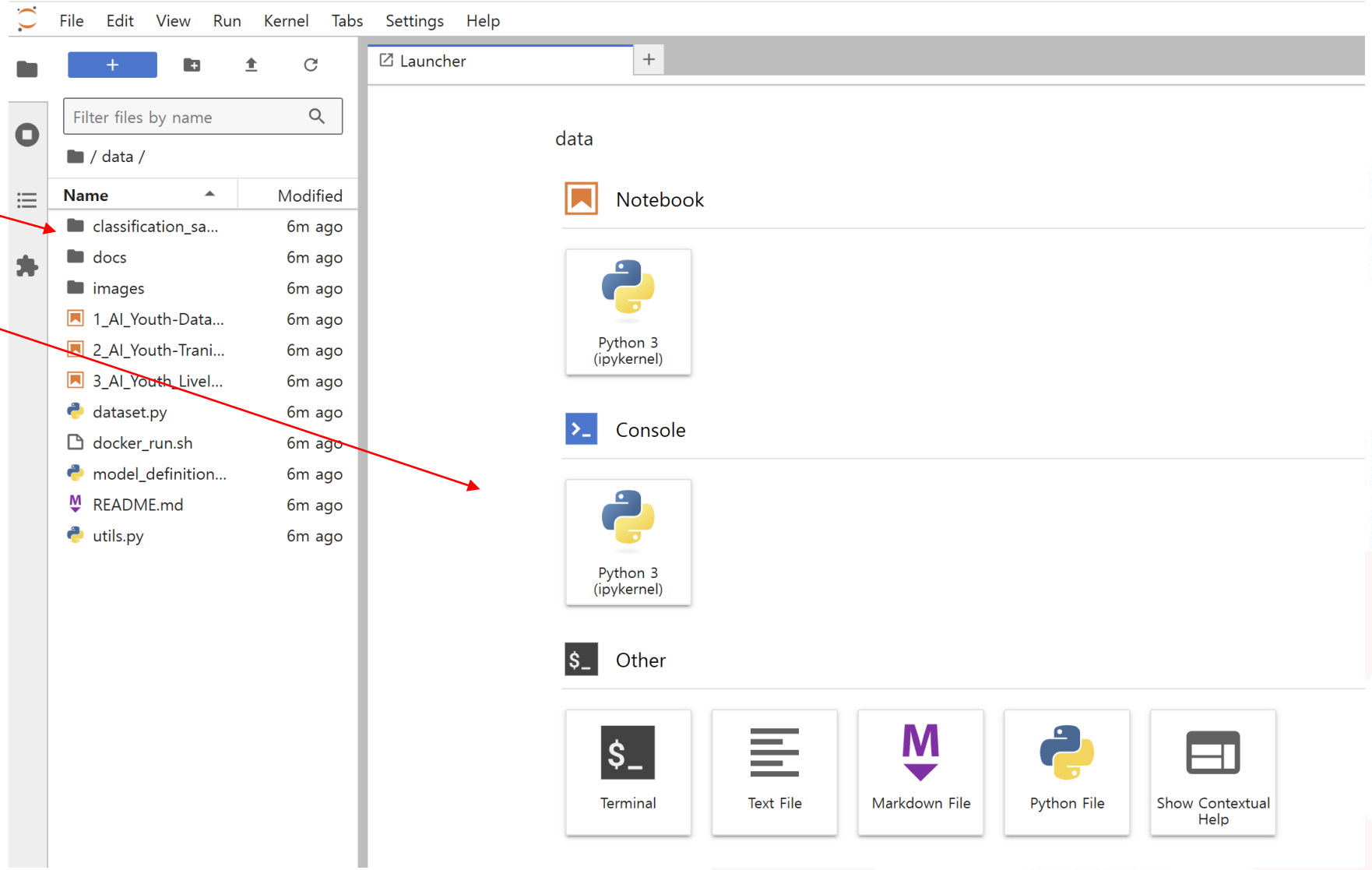
~/25_youth_winter : /nvdli-nano/data



  - 이미지 실행 후 /nvdli-nano/data 에 저장한 데이터는 도커 종료 후 ~/25_youth_winter 폴더에 남아있음

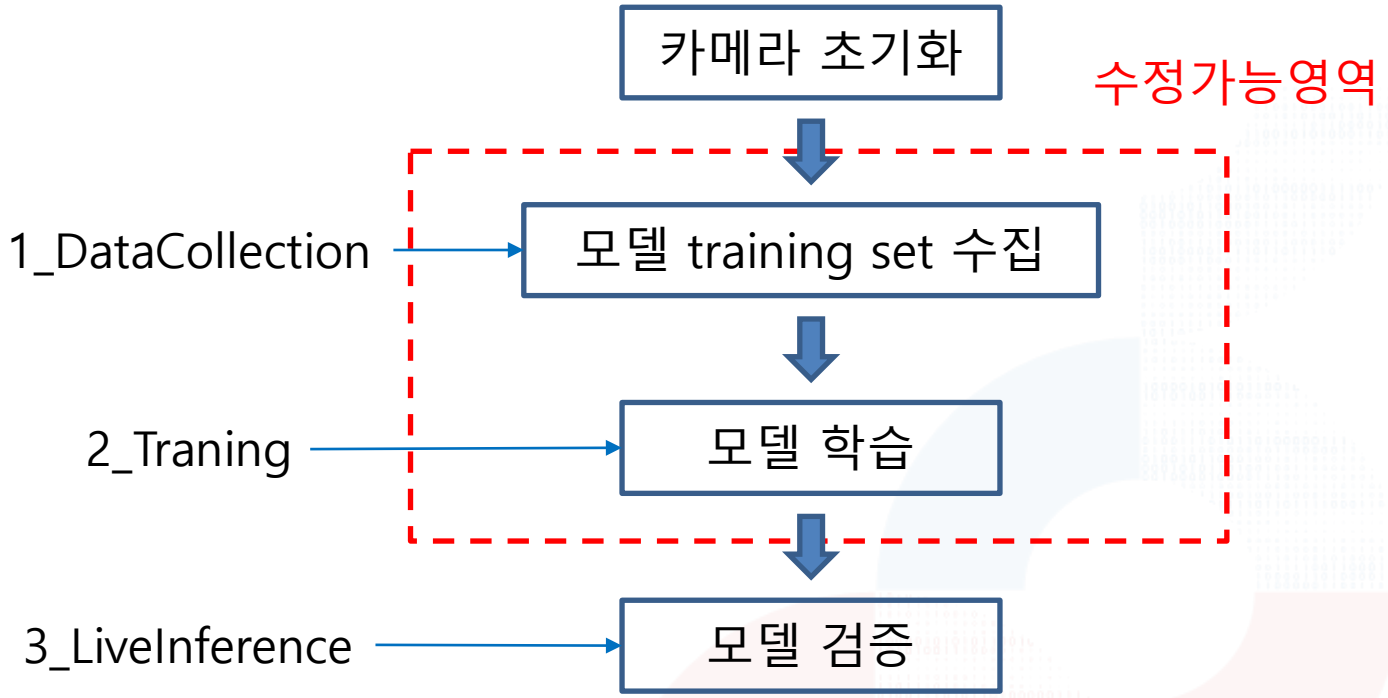
5.3 Jupyter Lab 환경

- Jupyter Notebook 이용
- 파일 경로 창
- 노트북 창



6. 딥러닝 실습 순서

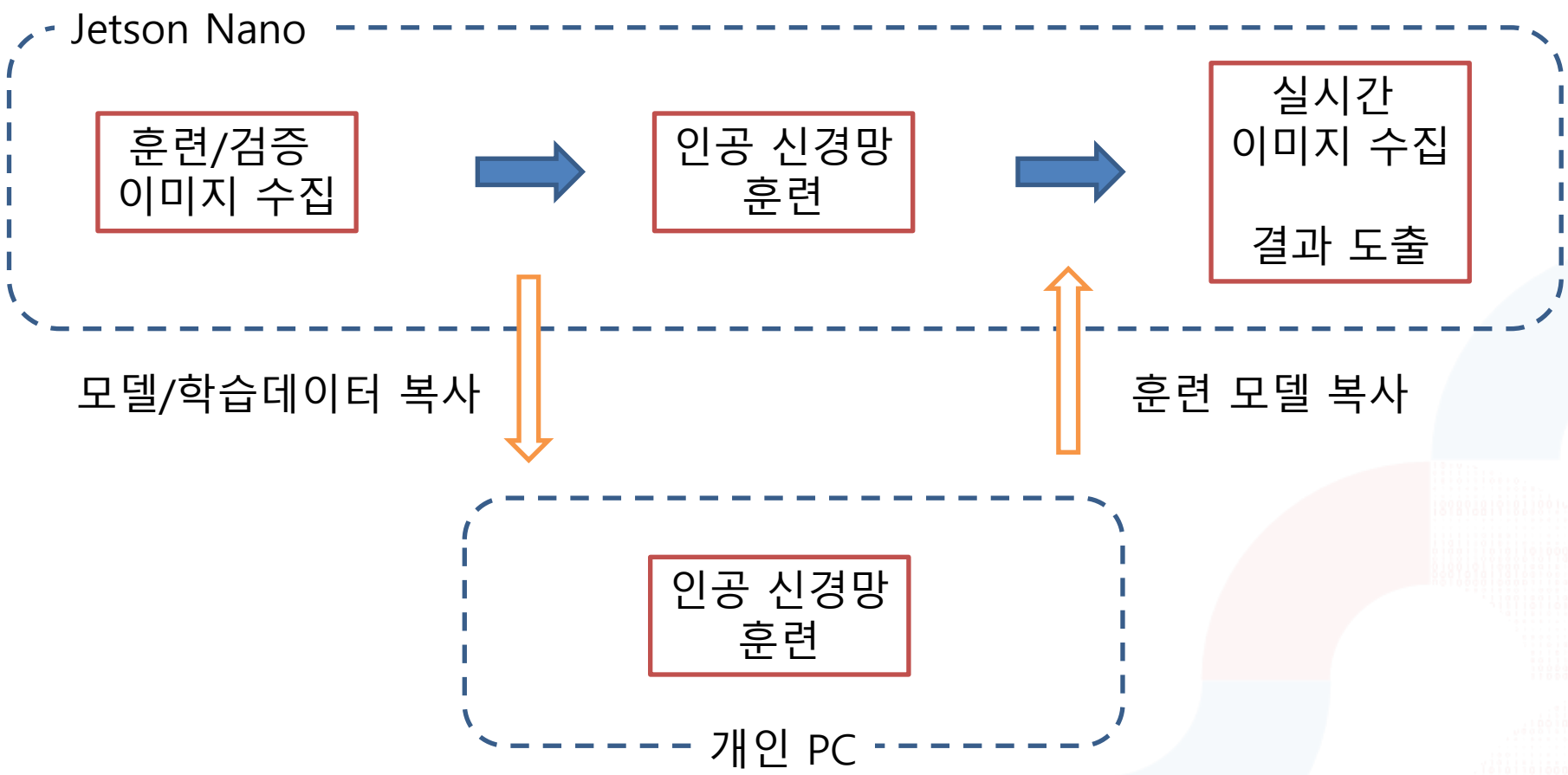
- 기본 개념
  - 신경망 구조 (CNN, RNN 등)
- 모델 훈련
  - 카메라를 이용한 데이터 셋 준비
  - 학습 및 검증
- 사전 훈련 모델 이용 가능
  - ResNet, MobileNet 등



- 학습 파일 다운로드 경로
  - https://github.com/jckim1201/25_youth_winter

6. 딥러닝 실습 순서

- 엄지 up/down 문제



학습데이터 수집

6.1 실습 코드 설명 - 학습 데이터 수집

- AI_data 폴더 내, 1_AI_Youth-DataCollection.ipynb 실행

- USB 카메라 연결

```
# USBCamera의 인스턴스는 한 번에 하나만 있을 수 있습니다. 이 셀을 시작하기 전에 실행되고 있는 카메라 인스턴스가 해제 되어있어야 합니다. 해제하기 위해 기존 인스턴스를 종료 하거나 커널을 종료해 주세요.
```

```
# jetcam 라이브러리를 이용한 USB 카메라 연결 설정
from jetcam.usb_camera import USBCamera

# usb 카메라 설정 (QVGA 사이즈, 가로:320, 세로:240 픽셀의 이미지를 video 0에서 획득)
camera = USBCamera(width=320, height=240, capture_device=0)

camera.running = True
print("카메라 확인 됨")
```

6.1 실습 코드 설명 - 학습 데이터 수집

- 문제 정의
  - 엄지가 위로 올라가거나 내려가는 움직임을 판단하여 구분을 하는 문제
  - 엄지가 위로 → thumbs_up
  - 엄지가 아래로 → thums_down

- 문제 설정

```
# 문제 제목 설정
TASK = 'thumbs'

# 문제 수집 데이터 정의
CATEGORIES = ['thumbs_up', 'thumbs_down']

# 수집 데이터 세트 정의 (각 조별)
DATASETS = ['A', 'B']

# 데이터 저장 폴더 지정, 만약에 없는 경우 이후 생성
DATA_DIR = './classification/'
```

추후 문제 값 제시 시, 제목과 데이터 정의 필요함

6.1 실습 코드 설명 - 학습 데이터 수집

- 이미지 저장 경로 설정

```
from dataset import ImageClassificationDataset

datasets = {}
for name in DATASETS:
    datasets[name] = ImageClassificationDataset(DATA_DIR + TASK + '_' + name, CATEGORIES)

print("{} 문제를 위한 {} 카테고리 정의 완료".format(TASK, CATEGORIES))

# 데이터 저장 폴더 지정, 만약에 없는 경우 생성
!mkdir -p {DATA_DIR}
```

- Data 수집 위젯 생성
  - 데이터 수집 도구 위젯을 생성하여 데이터 수집을 쉽게 할 수 있게 함

...

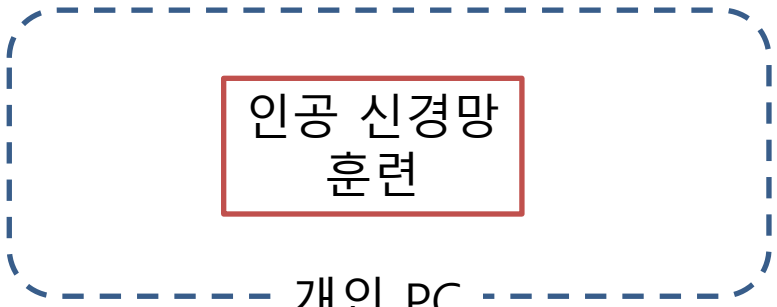
6.1 실습 코드 설명 - 학습 데이터 수집

- 실시간 데이터 수집
  - 파일이 저장 될 위치에 대한 dataset을 설정
  - 저장 할 category를 설정
  - 카메라에서 생성되는 이미지를 설정한 category에 맞도록 조정 후, add 버튼을 통해 그림 저장
  - 훈련 이미지 개수는 count에 나타나게 되며, 원하는 양에 맞게 계속 추가

- 카메라 및 프로그램 종료
  - 카메라 인스턴스 종료

6.1-2 수집 데이터의 복사

- 엄지 up/down 문제



FTP 프로그램 이용

IP : 각 팀 배치 jetson 주소

ID/PW: jetson/jetson
Data 폴더 명: ai-data

인공신경망 훈련

6.2 실습 코드 설명 – 인공 신경망 학습

- 2_AI_Youth-Traning.ipynb 실행
- 인공신경망을 이용한 이미지 훈련 및 검증
- 초기 입력 값 설정

```
# 훈련에 이용할 데이터 설정
data_dir = "./classification_sample/thumbs_A"

# 분류할 클래스 이름 ( 클래스 이름과 data_dir 에 위치한 폴더명을 맞춰야 함)
class_names = ['thumbs_down', 'thumbs_up']

# 하이퍼파라미터 설정
batch_size = 32
learning_rate = 0.001
epochs = 10

# 출력모델 이름 설정
Model_name = 'model_full.pth'

#-------------------------------------
# 이미지 크기 설정
image_width = 320   # 고정!!, 수정 금지
image_height = 240  # 고정!!, 수정 금지
```

6.2 실습 코드 설명 – 인공 신경망 학습

- 네크워크에 입력할 문제 정보 설정
  - 입력 값이 잘 되었는 지 확인
- 데이터 전처리
  - 데이터 증강, 이미지 크기 조정, 텐서 변환, 정규화 진행
- 이미지 데이터 입력
  - 수집한 이미지를 학습용, 검증용으로 구분하여 학습 시 이용
- 샘플 이미지를 통한 데이터 확인
  - 이미지와 category를 다시 한번 확인

6.2 실습 코드 설명 – 인공 신경망 학습

- 모델 설정 (model_definitions.py)
  - SimpleNN

- SimpleNN
  - 여러 계층을 순차적으로 진행하는 Sequential 모듈
  - 계층 구조
    - Fully connected
    - Linear type
    - ReLU(Rectified Linear Unit) 활성화 함수
      - 음수를 0으로 바꿔 비선형성을 추가
    - 각 단계에서 입력을 512→256→128로 줄이는 선형 계층
    - 마지막 출력 단에서 num_classes 만큼 출력
  - Forward method
    - 입력데이터를 2D 형태로 변환, 다차원 데이터를 신경망 계층에 맞게 평탄화 하는 작업
    - nn.sequential로 정의된 계층을 통과시켜 결과를 반환

```
import torch.nn as nn

# 기본 신경망 클래스 정의
#
class SimpleNN(nn.Module):
    def __init__(self, input_size, num_classes):
        super(SimpleNN, self).__init__()

        self.layers = nn.Sequential(
            nn.Linear(input_size, 512),
            nn.ReLU(),

            nn.Linear(512, 256),
            nn.ReLU(),

            nn.Linear(256, 128),
            nn.ReLU(),

            nn.Linear(128, num_classes)
        )

    def forward(self, x):
        x = x.view(x.size(0), -1)
        x = self.layers(x)
        return x
```

6.2 실습 코드 설명 – 인공 신경망 학습

- 모델 가중치 초기화, 손실 함수 및 최적화 함수 설정
- CPU, GPU 이용 확인
- 학습/검증 과정에 대한 함수 설정
- 훈련 시작

```
Epoch [1/10] Train Loss: 0.7492, Train Accuracy: 48.48% | Val Loss: 107.0017, Val Accuracy: 55.56%
------------------------------
Epoch [2/10] Train Loss: 13.9962, Train Accuracy: 69.70% | Val Loss: 2.4473, Val Accuracy: 77.78%
------------------------------
Epoch [3/10] Train Loss: 0.1696, Train Accuracy: 87.88% | Val Loss: 5.2279, Val Accuracy: 55.56%
------------------------------
.....
------------------------------
```

- 검증용 샘플 확인 ➔ 모델 저장 (model_group#.pth) : 과제 후 해당 파일 제출

!! 예) 1조인 경우 : “model_1.pth” 로 저장

실시간 훈련 모델 확인

6.3 실습 코드 설명 – 실시간 훈련 모델 확인

- 3_AI_Youth_LiveInference.ipynb 실행
- USB 카메라를 이용한 실시간 훈련 모델 확인
- 입력 값 설정

```
# 모델 경로와 클래스 정보 설정

MODEL_PATH = '/nvdli-nano/data/model_1.pth'

CATEGORIES = ['thumbs_down', 'thumbs_up']  # 카테고리 목록
```

- 모델 확인

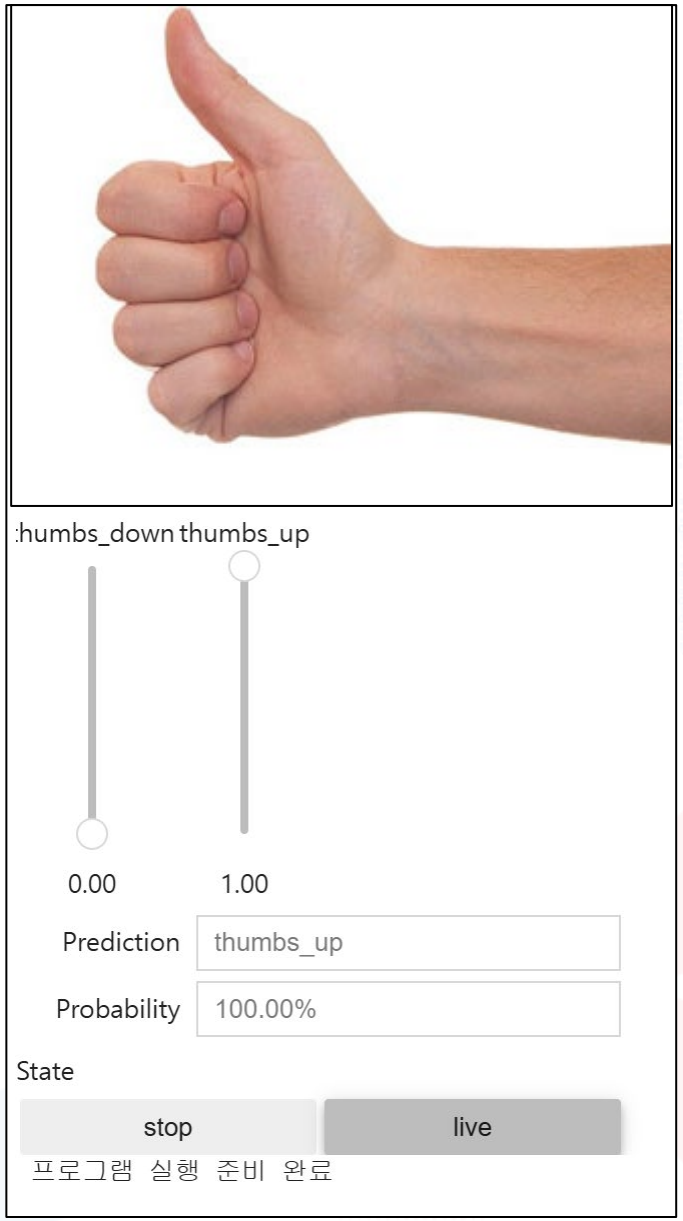
```
model = torch.load(MODEL_PATH)
```

- 모델 결과를 위젯을 통해 확인

6.3 실습 코드 설명 – 실시간 훈련 모델 확인

- 위젯 설명
  - 훈련이미지 수집과 유사
  - 모델을 읽어 들인 후, live 버튼 작동 시, 모델이 실행 됨
  - 실시간으로 이미지의 예측 결과가 도출 됨
  - 예측 결과의 확률과 구분 값을 바 그래프와 값으로 출력

- 카메라 인스턴스 종료 및 커널 종료



참고 자료

Appendix.1 기타 참고 사이트

- Jetson AI Fundamentals - S1E3 - Image Classification Project

  (https://www.youtube.com/watch?v=rSqIvLQ8Meg&ab_channel=NVIDIADeveloper)

- https://www.jetson-ai-lab.com/

- https://developer.nvidia.com/embedded/community/jetson-projects

- https://github.com/jckim1201/25_youth_winter

Appendix.2 Desktop: Conda pytorch 개발 환경 설정

- Pytorch 환경 설치 (conda base)

```
$conda env list
$conda create -n ai_pytorch
$conda activate ai_pytorch
$conda install pytorch torchvision torchaudio cpuonly -c pytorch
$conda install ipython
```

- Jupyter notebook 환경 설치 및 pytorch 가상환경 연결

```
$pip install jupyter notebook
$python -m ipykernel install --user --name ai_pytorch --display-name ai_pytorch
```

```
$jupyter kernelspec list
$jupyter kernelspec uninstall ai_pytorch
```

과제 및 제출방법

Appendix.3 평가 과제

- **묵,찌,빠(Rock, Paper, Scissors)** 를 위한 분류 모델 만들어 보기
- 모델의 layer 층, layer 노드 수, 활성함수, epoch 수 등을 수정하여 모델을 개선하기
- 2_AI_Youth_Traning.ipynb 파일을 이용
- Jetson Orin Nano 또는 개인 PC에서 훈련을 진행
- 훈련된 모델을 **23일(목) 오후 9시(21:00)까지 jckim1201@kisti.re.kr 로 제출**
  - **회신 메일**을 못 받은 경우, 다시 한번 제출 확인 필요!!
  - 메일 제목 : **"2025 청소년캠프 – 1조 결과 제출"**

  다음의 두 파일을 반드시 보내야 함
  - **Model_definitions.py**
  - **model_(group #).pth**