



👑 Team Leader: EunJi Jung  
🚀 App Developer: EunSeong Lee  
🌟 OnDevice Developer: HyenWoo Choi  
💡 Validation Engineer: J. Hwan

조은지안조은지조 | Feat.DPU에 따른 HW 성능 비교

# Content

## 01 개요

- AI를 활용한 영양제 추천 시스템
- 주제 선정 배경
- 개발 환경

## 02 학습 모델 개발

- Introducing Dataset & Model
- Loss & Accuracy 성능 평가
- 웹캠 연동

## 03 모델 통합 및 결과

- Ollama Model
- S/W Architecture
- Flow Chart
- Main Features

## 04 가속기 w FPGA

- FPGA DPU 구현 과정
- Performance comparison-1: ResNET50
- Performance comparison-2: YoloV3

## 05 Trouble Shooting

- Trained AI Model
- Integrated System
- Accelerator

## 06 시연 및 고찰

- 시연 영상
- 고찰
- Q&A



01 | 개요

# 서비스 개요



01

본 프로젝트는 카메라를 기반으로 안면인식을 하여 나이와 성별을 추론하여, 사용자의 적합한 맞춤형 영양제를 추천하는 인공지능 디지털 헬스케어 시스템을 제작한 Nutrfit 솔루션입니다.

## 서비스 필요

01

개인 맞춤형 건강 관리에 대한 수요가 급증함으로써 서비스 구축

02

과도한 마케팅으로 인한 불필요한 영양제 구매 및 섭취 문제

03

개인별 건강 상태 및 생활 습관에 대한 정확한 정보 부족

# Develop Environment

**HW**



Edge Device  
(Raspberry Pi)

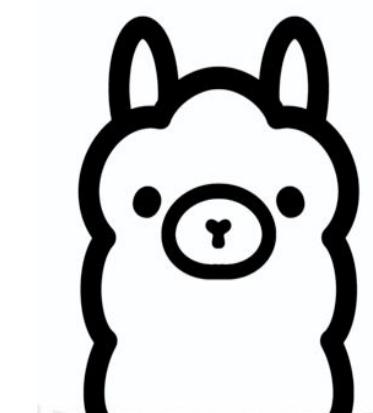
FPGA  
(Ultra96-V2)

Webcam  
(Pleomax 300K)

**SW**



Python



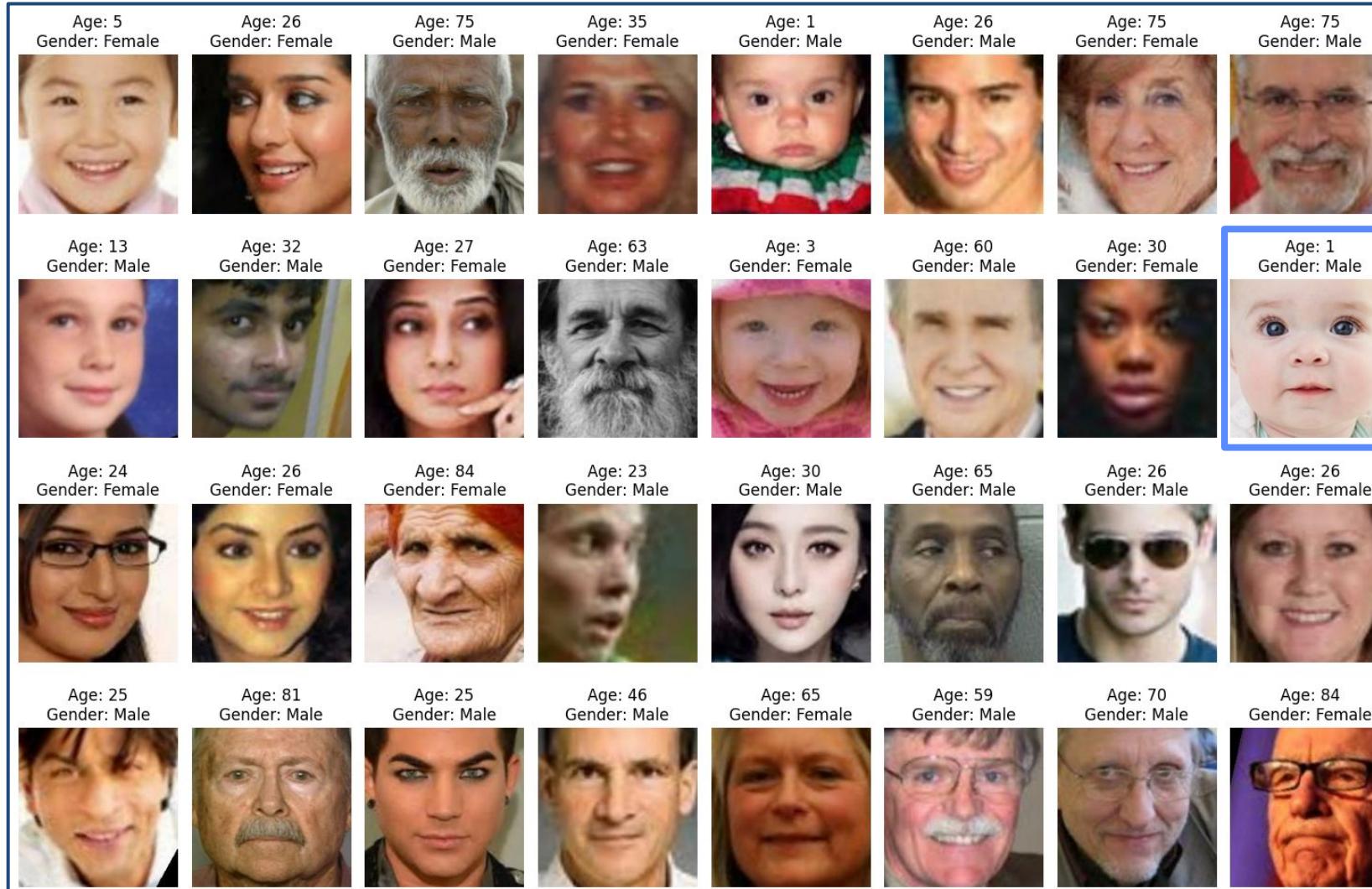
LLM  
(Ollama)



## 02 | 학습 모델 개발

# DataSet: UTKFace

UTKFace 데이터란? 얼굴 관련 모델을 훈련하고 평가하는 데 사용되는 대규모 공개 데이터셋



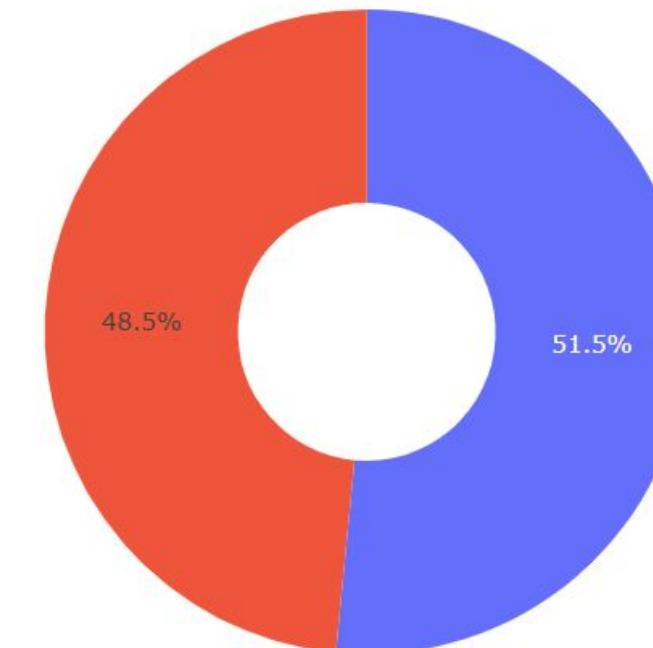
1 \_ 0 \_ 0 \_ 20161219225952240.jpg

↓  
나이 성별 인종  
수집 날짜 및 시간

분류	문제 유형	예측값	출력층	Loss Func.
나이	회귀	실수	Dense(1)	MAE MSE
성별	이진 분류	확률 0 or 1	Dense(1, activation = 'sigmoid')	binary_crossentropy

# DataSet: UTKFace

UTKFace 데이터 출처: Kaggle, Roboflow

	Age	Gender	성비
데이터 개수	2,087	10,284	
Training Size	1,460	8,227	
Validation Size	417	1,028	
Testing Size	210	1,029	

# Backbone Model

Backbone이란? 이미지로부터 특징을 추출하는 역할을 하는 CNN Network

AgeNet

Layer (type)	Output Shape	Param #
VGG16( <a href="#">Functional</a> )	(None, 7, 7, 512)	14,714,688
Slight Dropout ( <a href="#">Dropout</a> )	(None, 7, 7, 512)	0
FlattenEmbeddings( <a href="#">Flatten</a> )	(None, 25088)	0
dense ( <a href="#">Dense</a> )	(None, 256)	6,422,784
AgeOutput ( <a href="#">Dense</a> )	(None, 1)	257

GenderNet

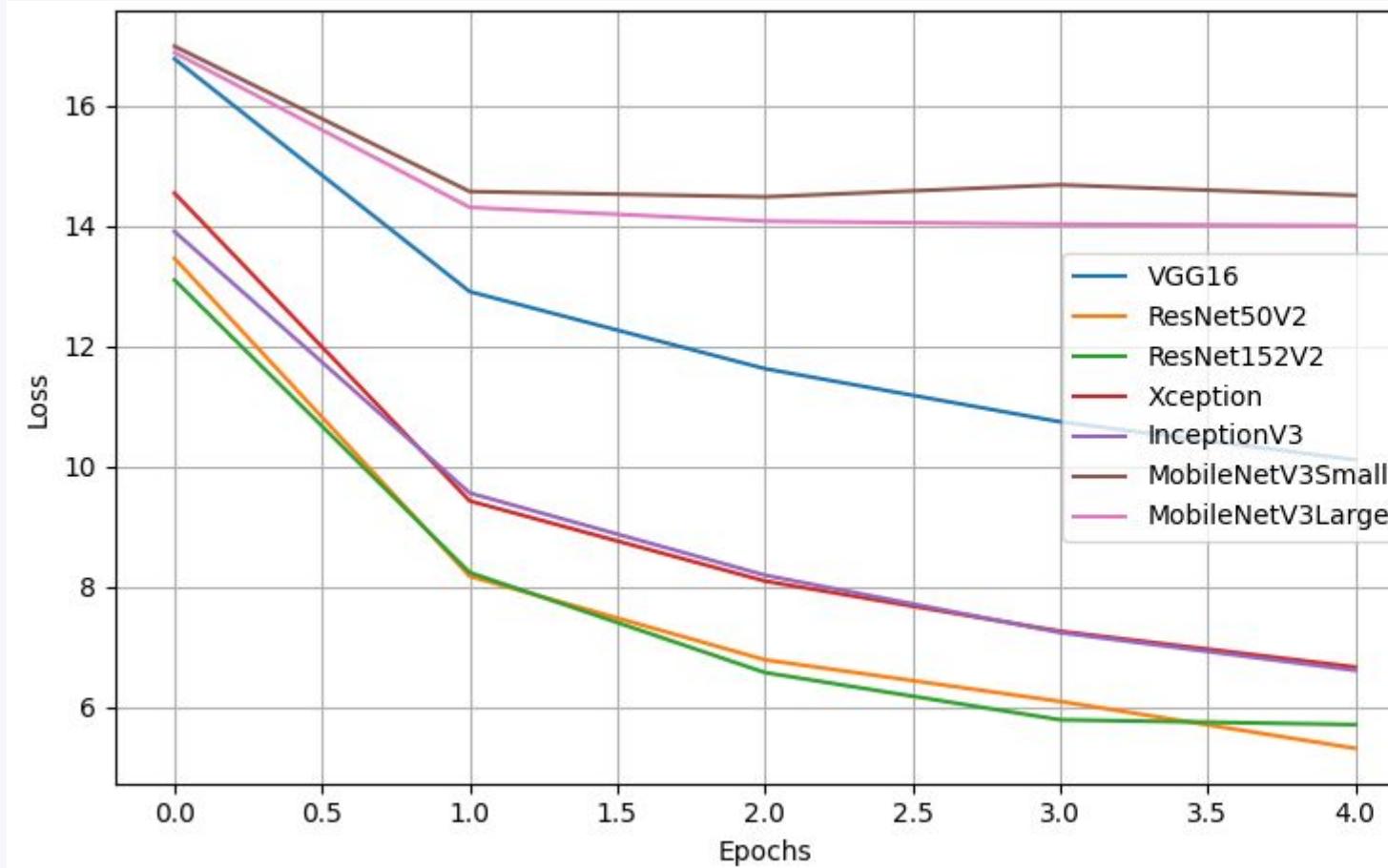
Layer (type)	Output Shape	Param #
ResNet152V2( <a href="#">Functional</a> )	(None, 7, 7, 2048)	58,331,648
Slight Dropout ( <a href="#">Dropout</a> )	(None, 7, 7, 2048)	0
GlobalAvPooling ( <a href="#">GlovalAveragePooling2D</a> )	(None, 2048)	0
gender ( <a href="#">Dense</a> )	(None, 1)	2,049

\* Batch Size = 32, Epochs = 20, Optimizer = adam, Dropout Rate = 0.2 / 0.4

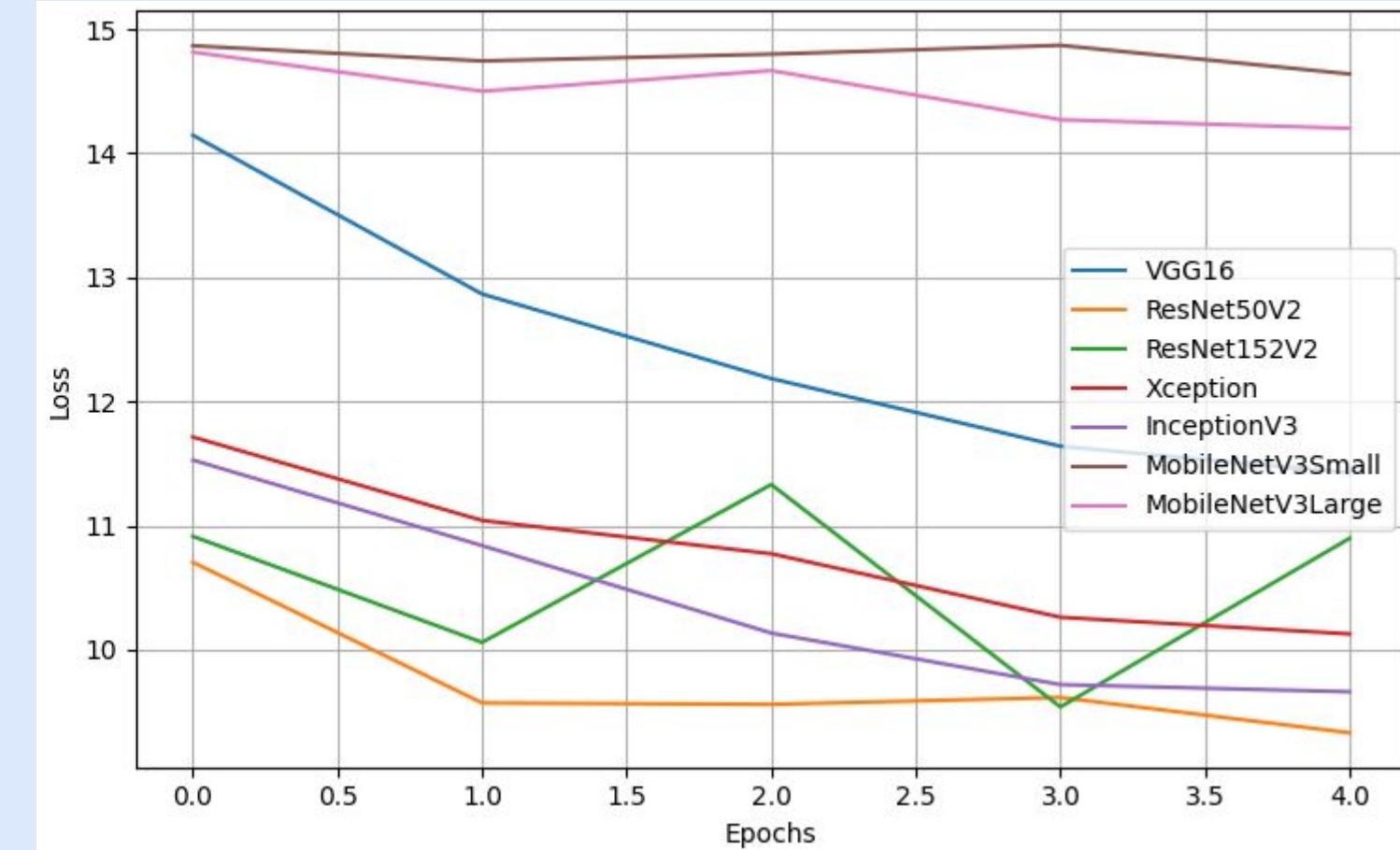
# Backbone Model

AGE Loss plot Backbone Model Comparison

Train Loss Plot



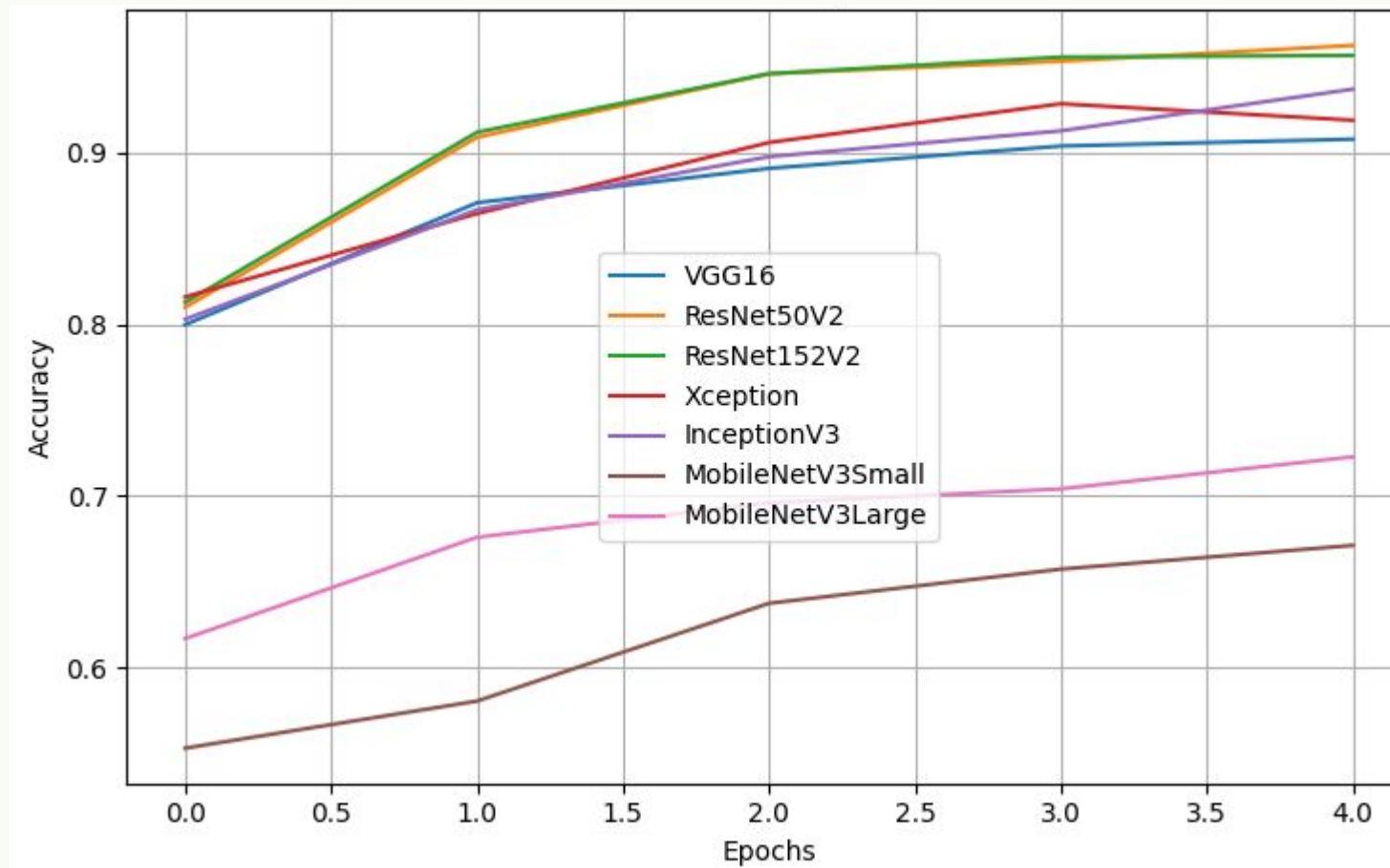
Val Loss Plot



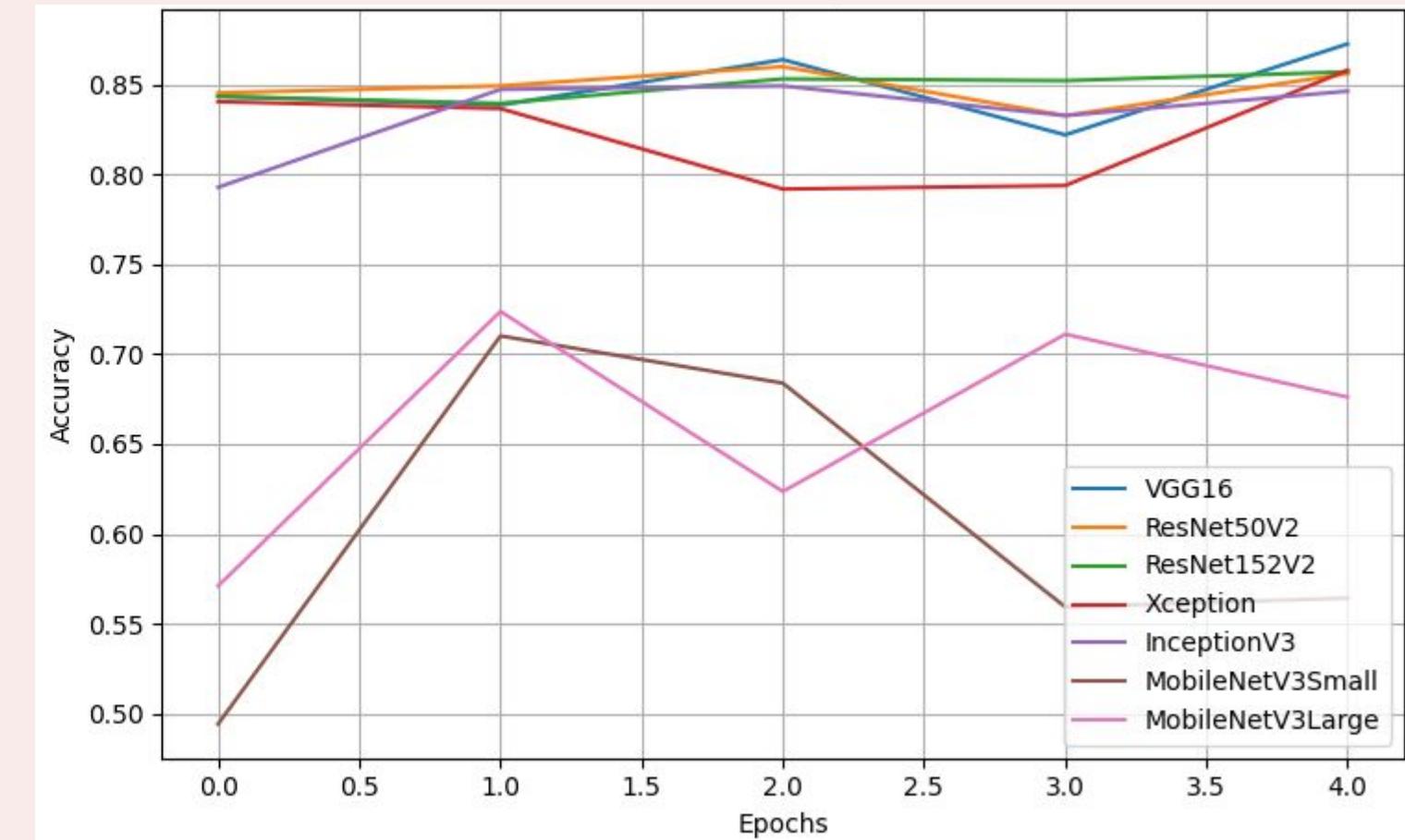
# Backbone Model

Gender Accuracy plot Backbone Model Comparison

Train Accuracy Plot



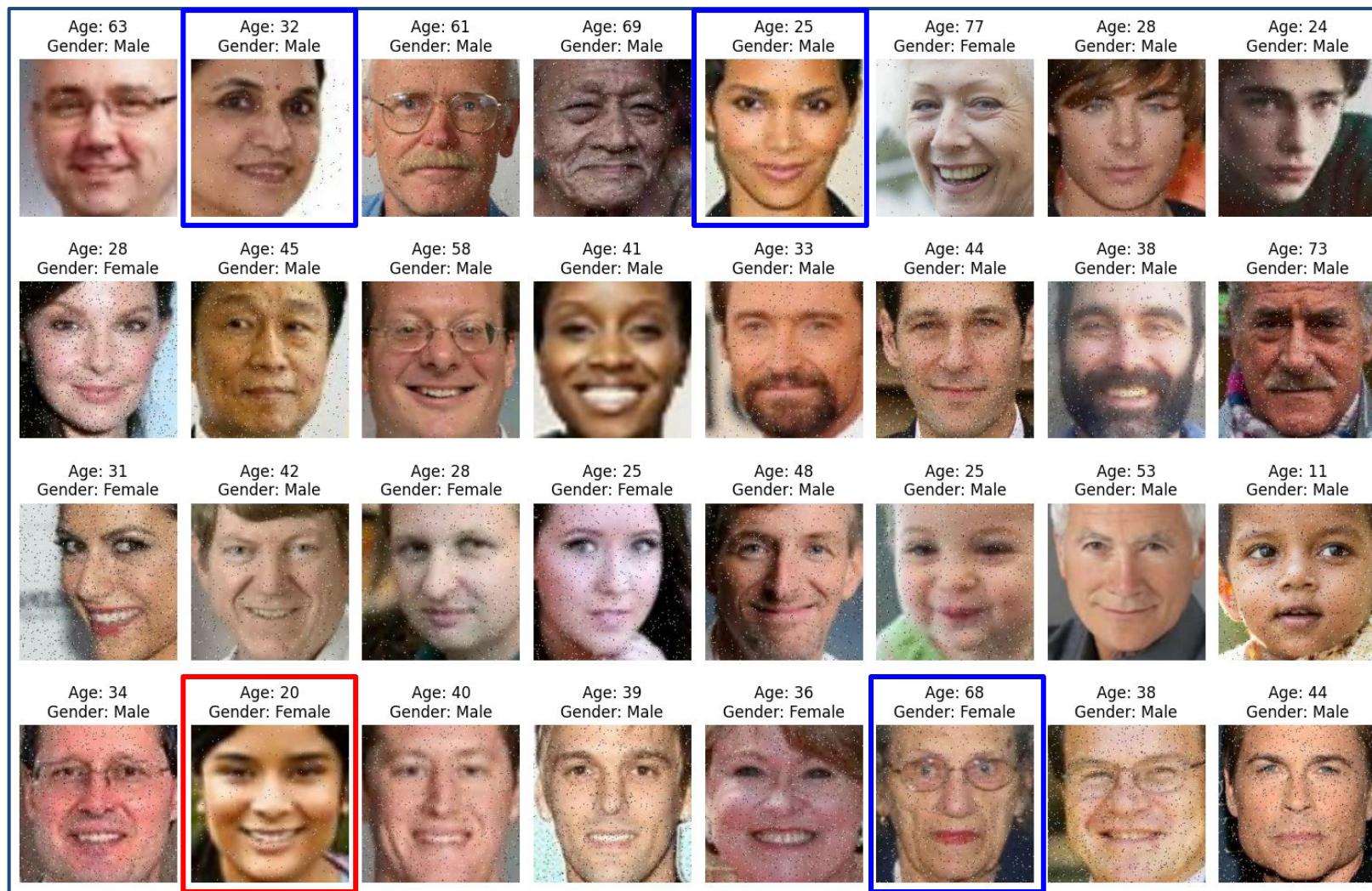
Val Accuracy Plot



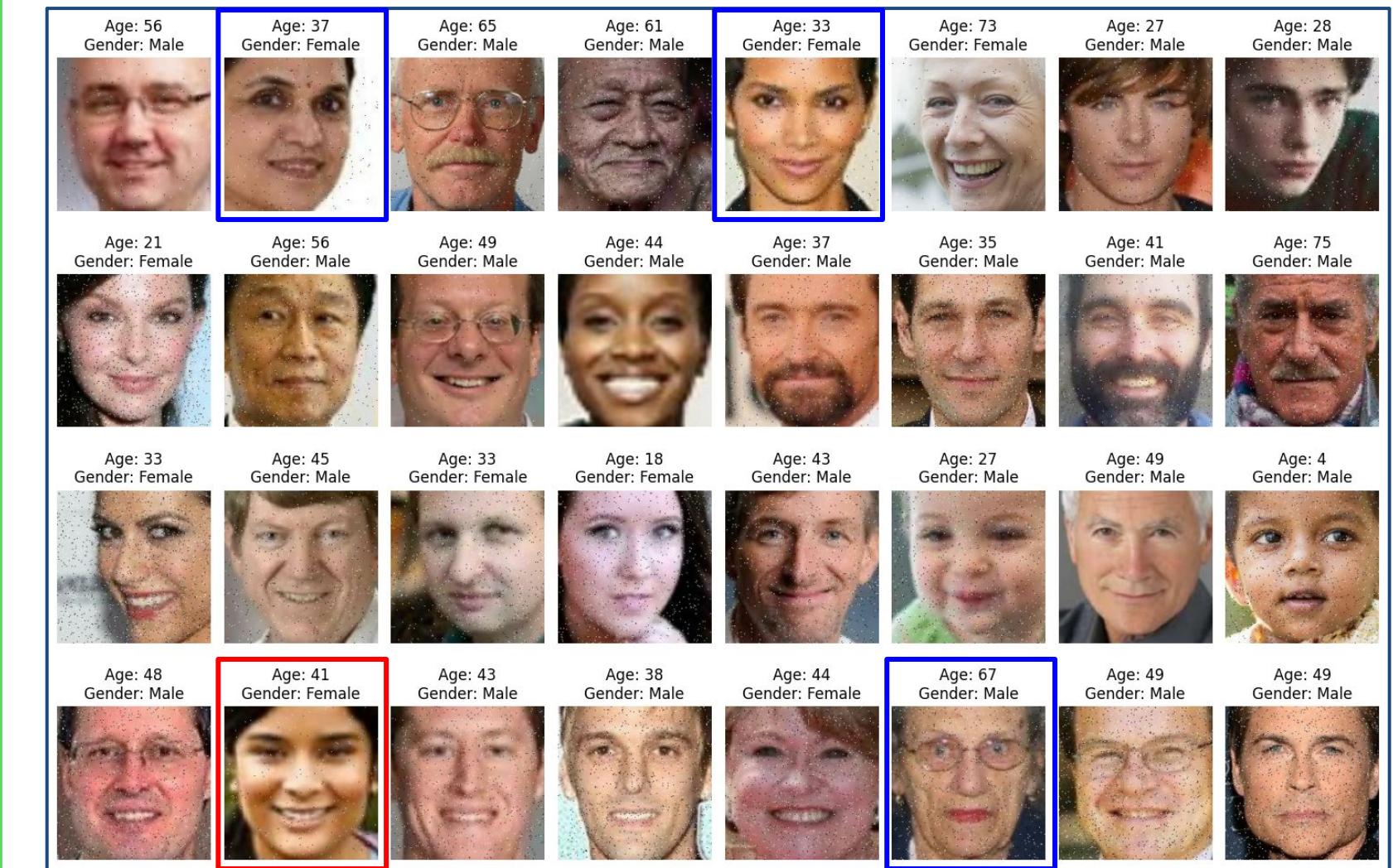
# Model Prediction

Age Backbone Model / Gender Backbone Model

VGG16 / ResNet152V2

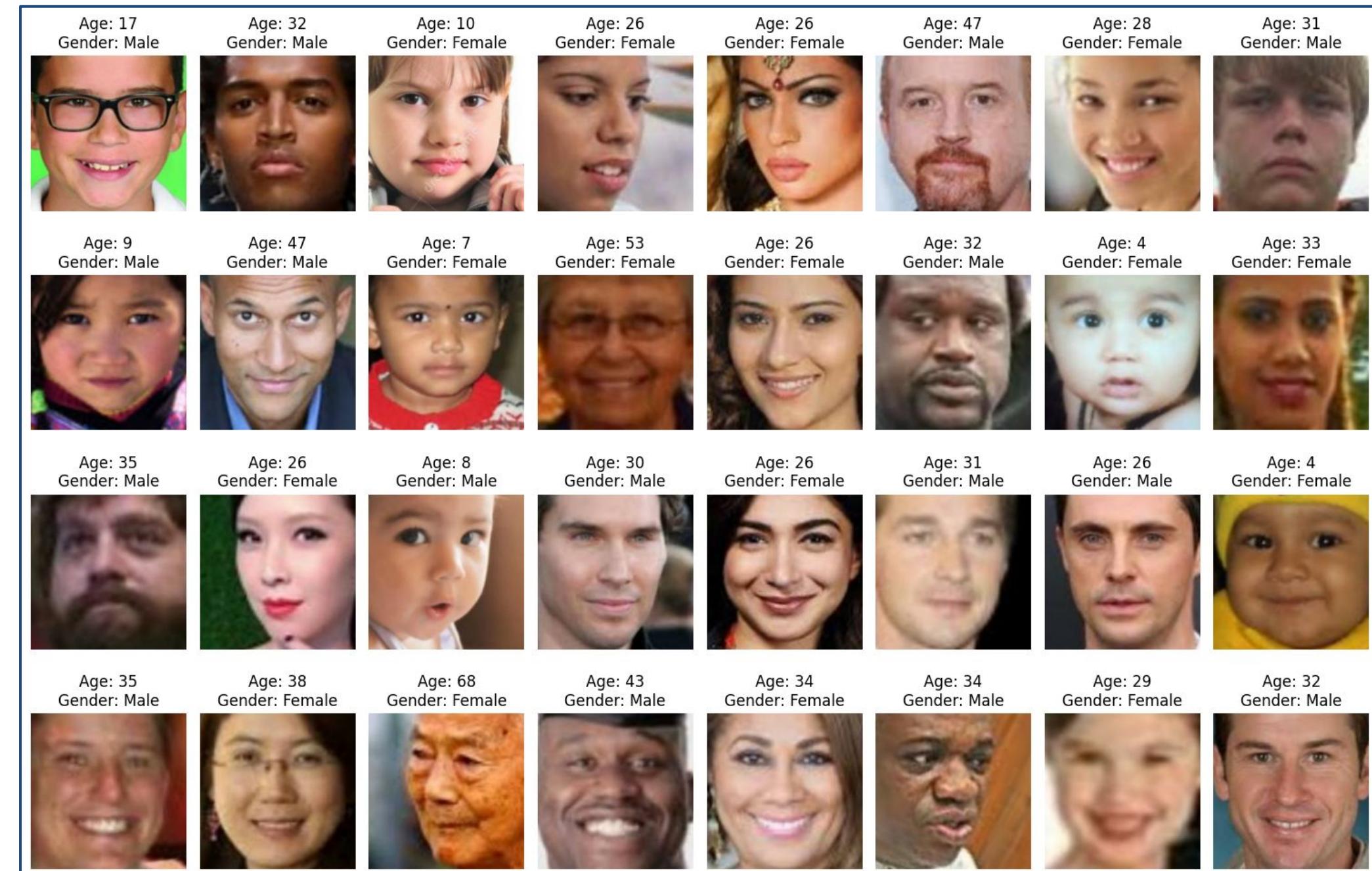


ResNet50V2/Xception



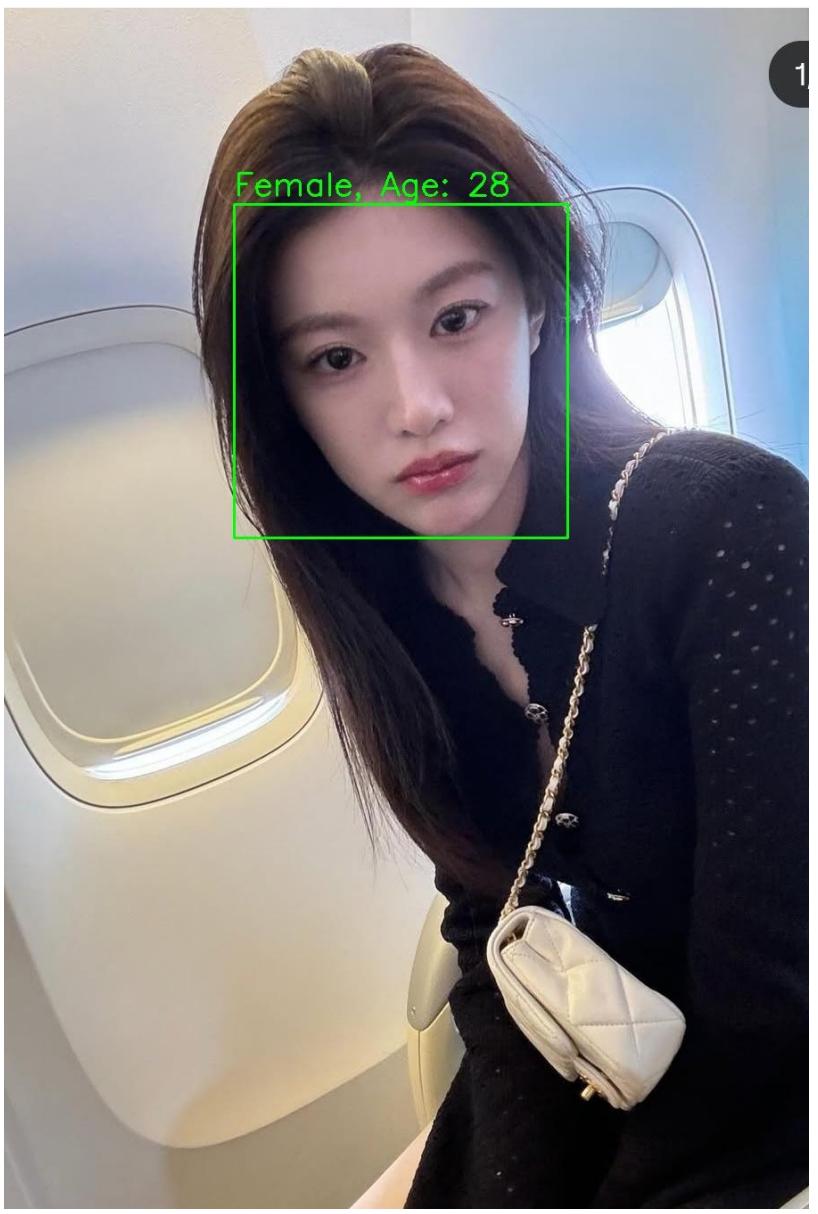
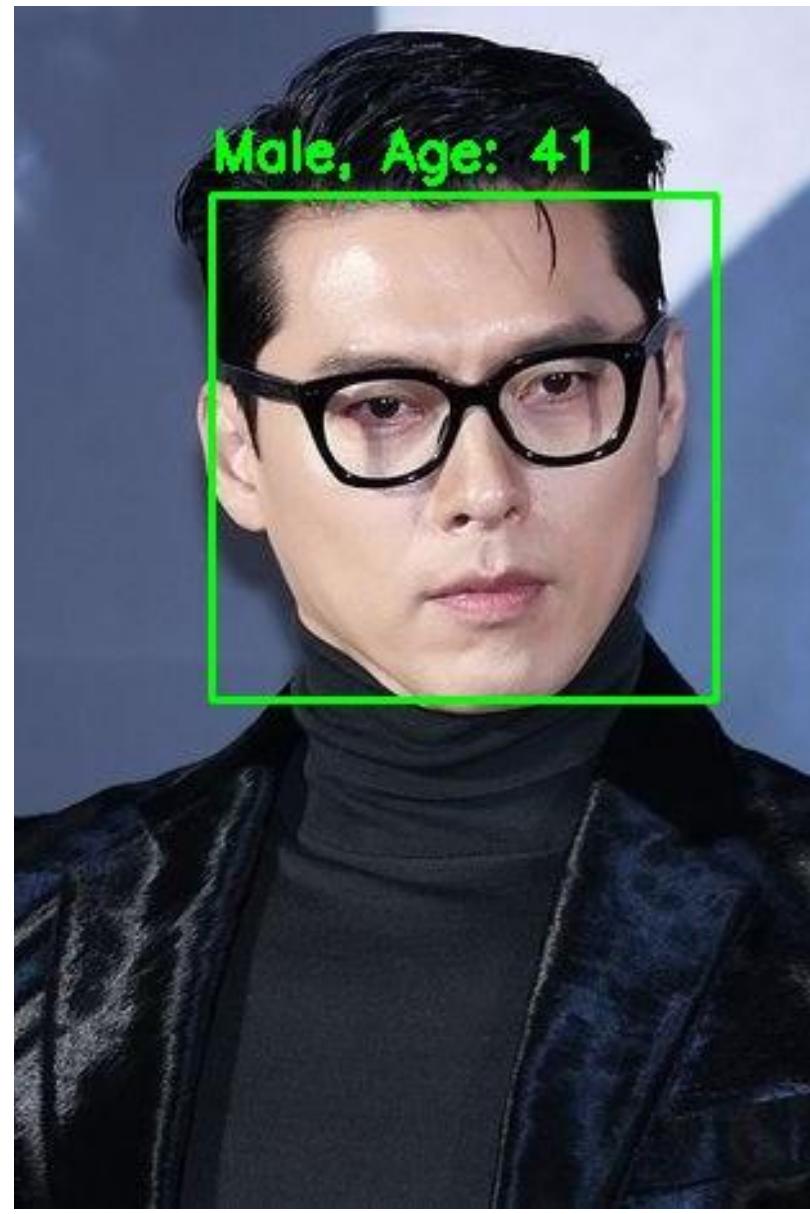
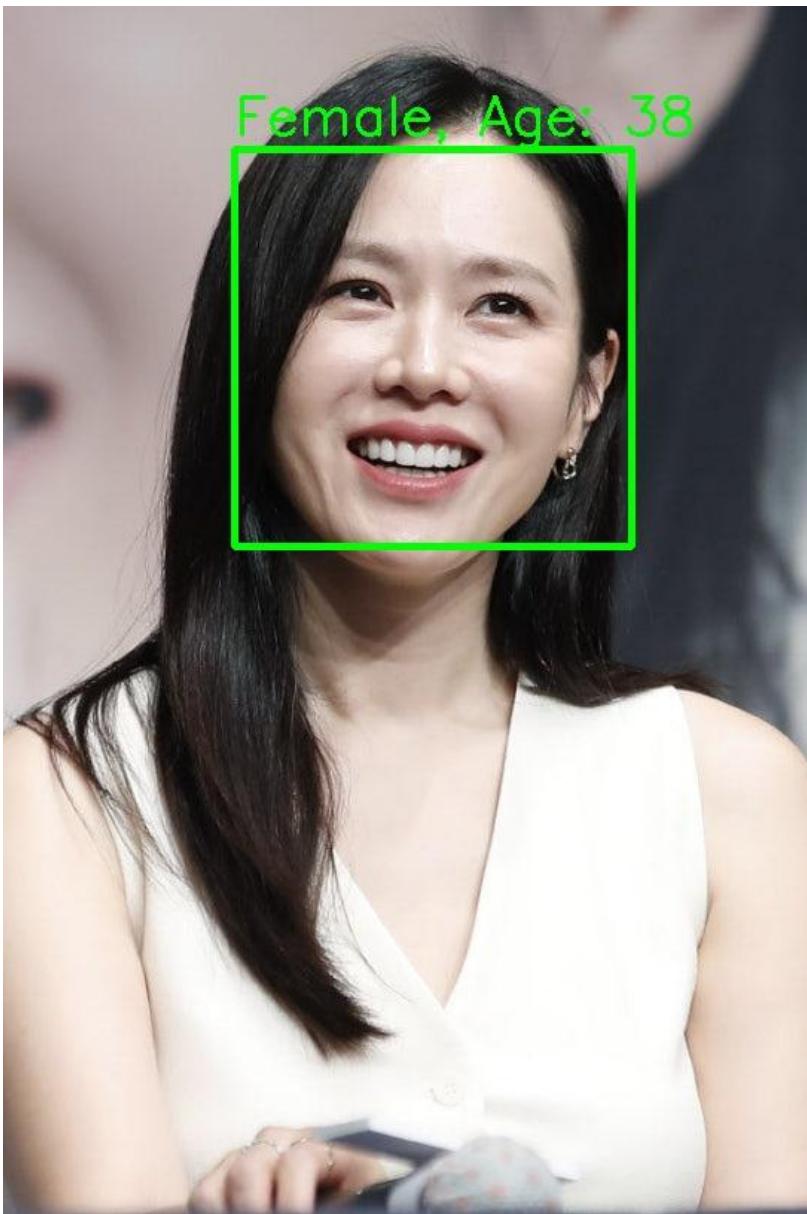
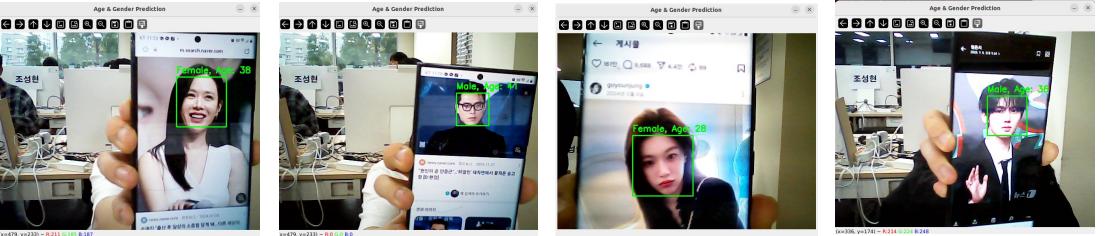
# Model Prediction

Data 개수 증가 ⇒ 성별 정확도 증가



# Model Prediction

WebCam Prediction



손예진(여/만 43세)

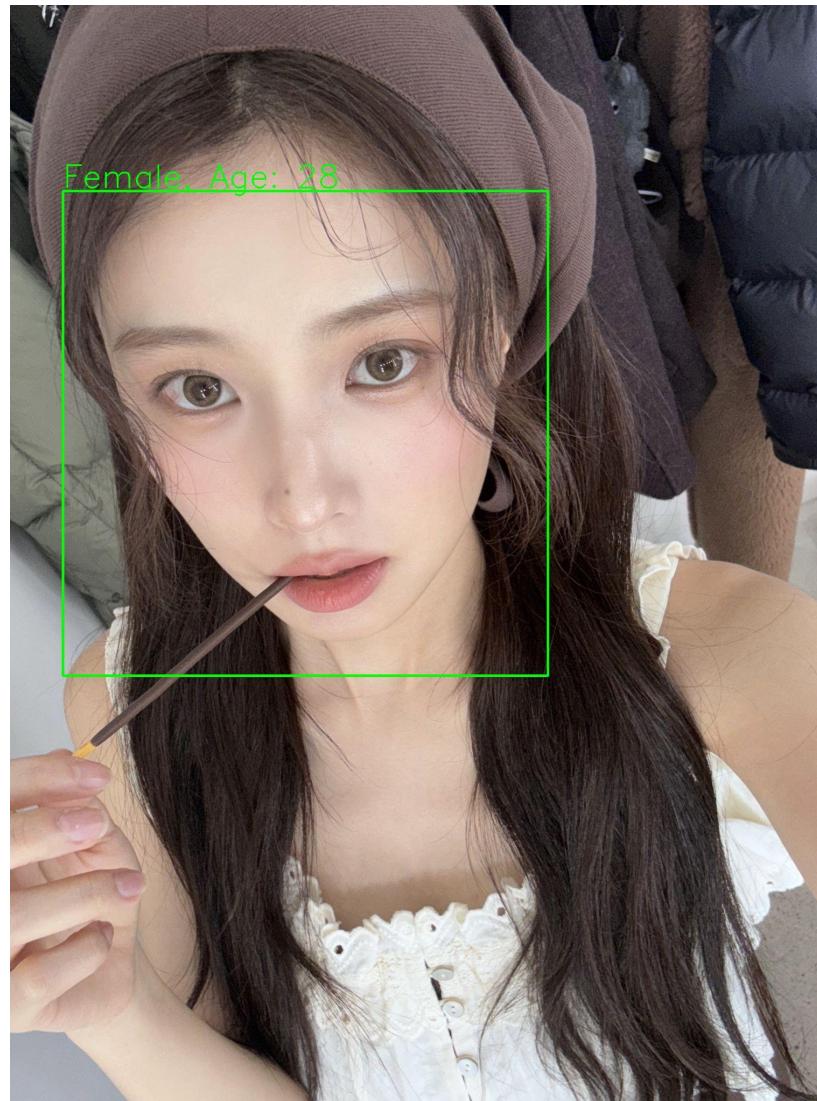
현빈(남/만 42세)

고윤정 (여/만 28세)

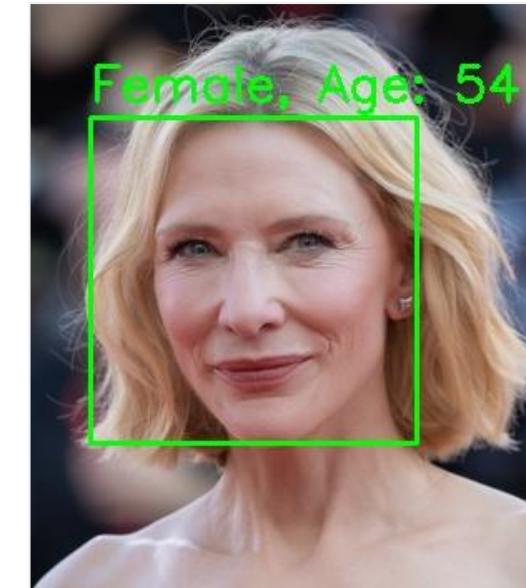
한유진 (남/만 19세)

# Model Prediction

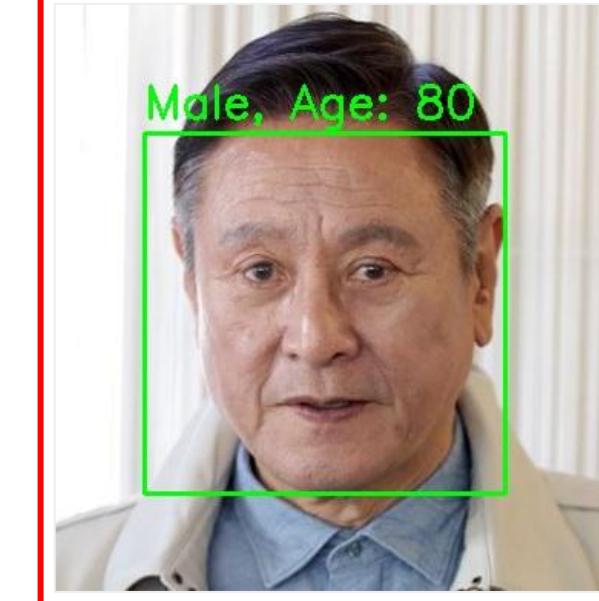
Image Prediction: 동서양 및 남녀 학습도 비교



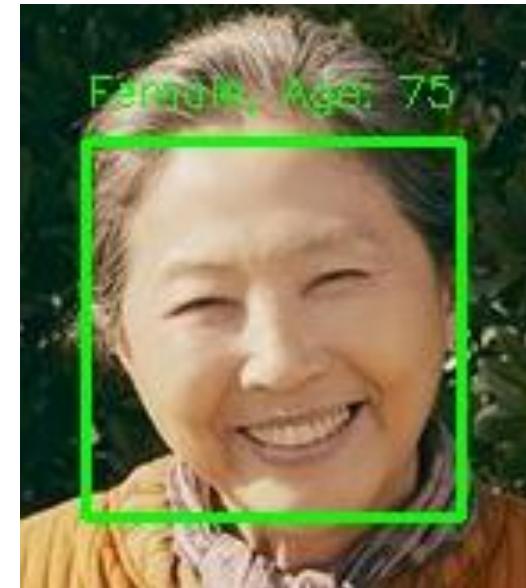
강혜원(여/만 27세)



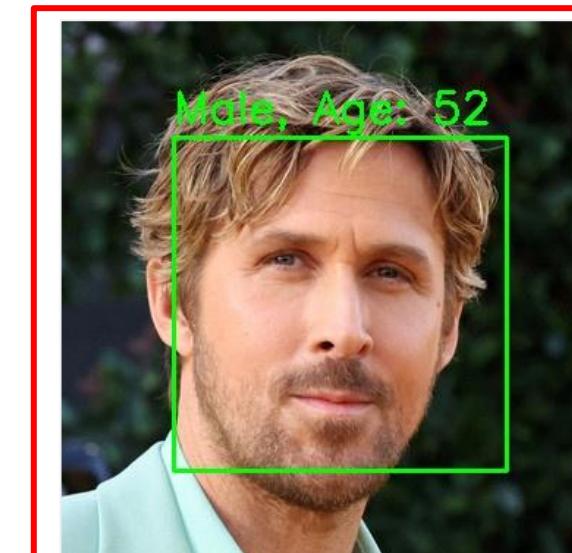
케이트 블란쳇(여/만 55세)



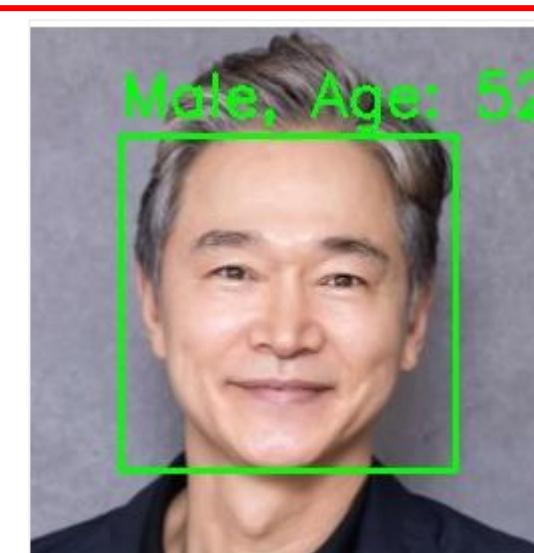
박근형 (남/만 75세)



고두심 (여/만 71세)



라이언 고슬링(남/만 43세) 정보석 (남/만 62세)



톰 행크스 (남/만 62세)

# Model Prediction

추론 코드 후보정 처리

## WebCam Prediction

- 후보정 적용 X
- 웹캠의 실시간 데이터  $\Rightarrow$  데이터 증강 효과
- WebCam 추론 성능  $>$  Image 추론 성능

## Image Prediction

\* 후보정

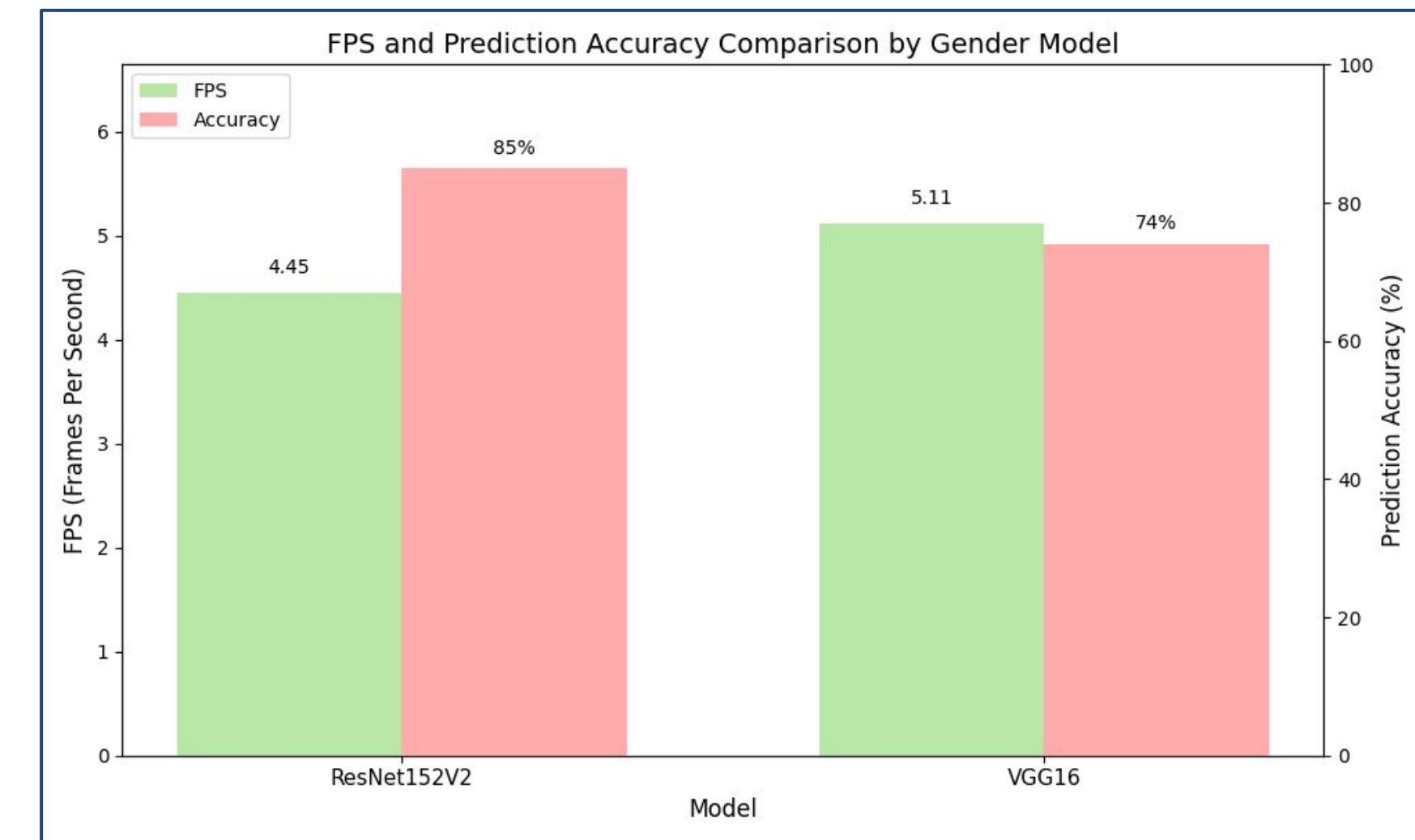
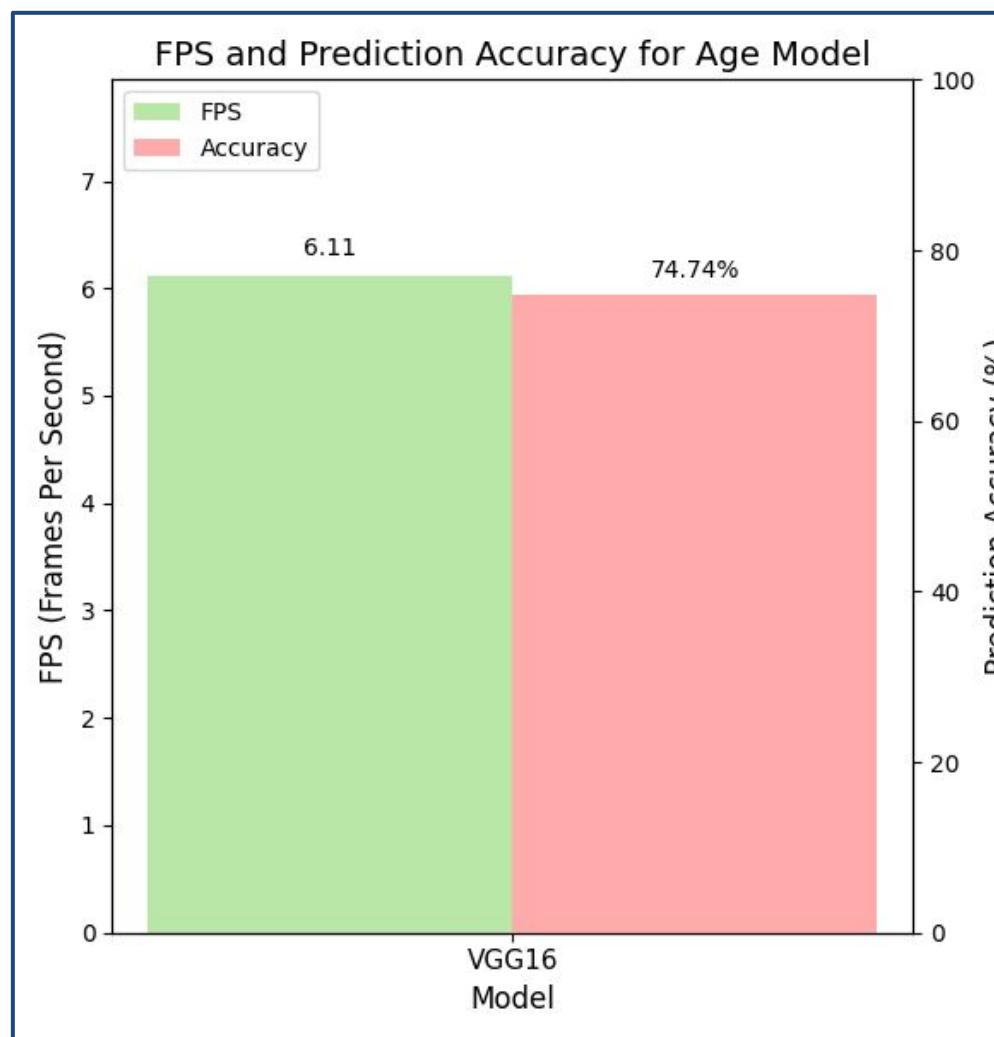
20-30	30-40	40-50	50-60	60-70	70-80
- 7	- 4	+ 5	+ 8	+ 13	+ 18

- 서양인 DataSet 多  $\Rightarrow$  서양인에 더 적합한 모델
- 이미지 추론 후보정 적용  $\Rightarrow$  기준 나이 대비 서양인: / 동양인:
- WebCam, Image 추론 모두 여성 대비 남성의 나이 판단

# Model Prediction

Test UTKFace Data

- 일반 데이터가 아닌 UTKFace Data로 예측 정확도 추론 성능 비교

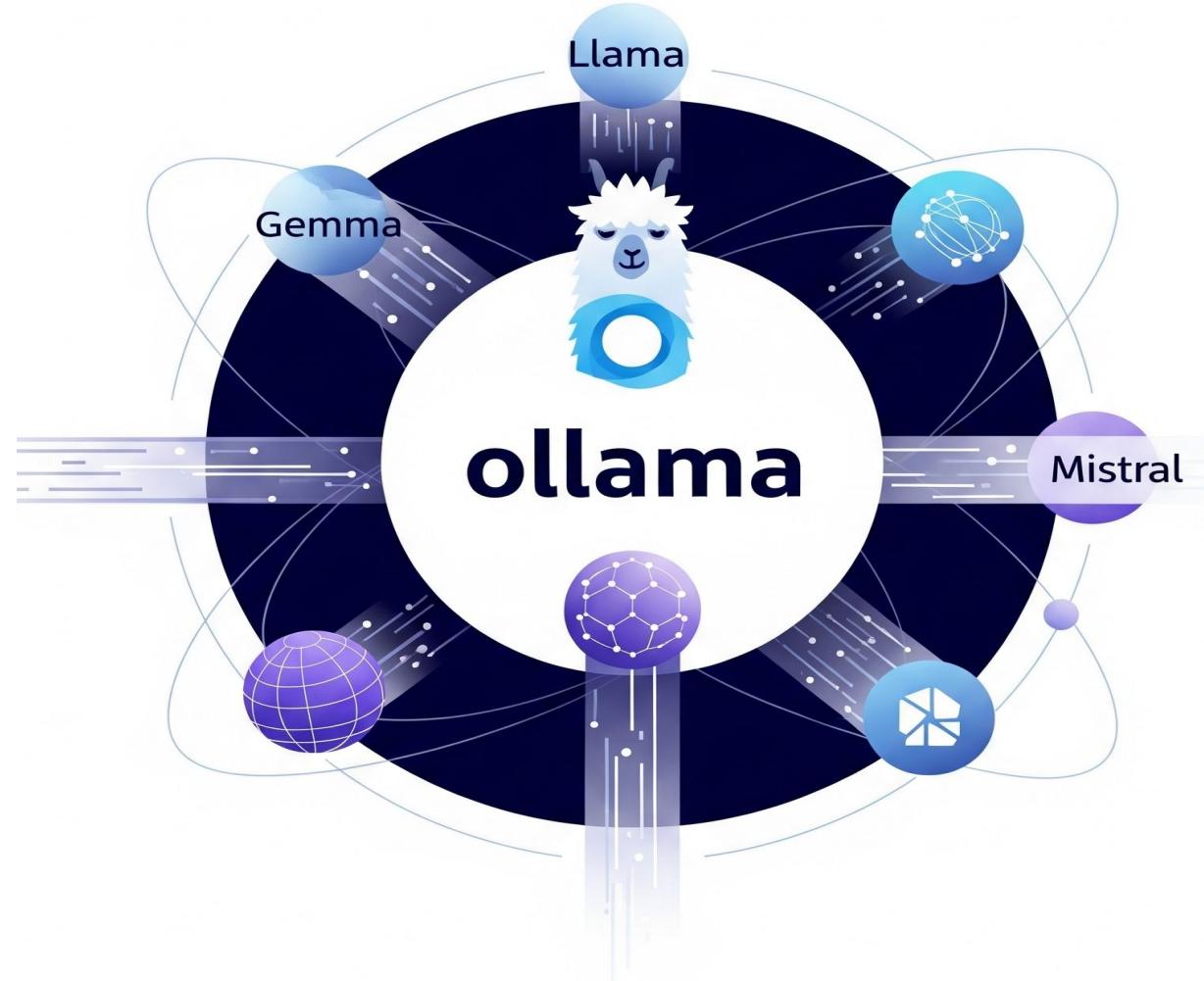




03 | 모델 통합 및  
결과

# Ollama Model

---



## Ollama ?

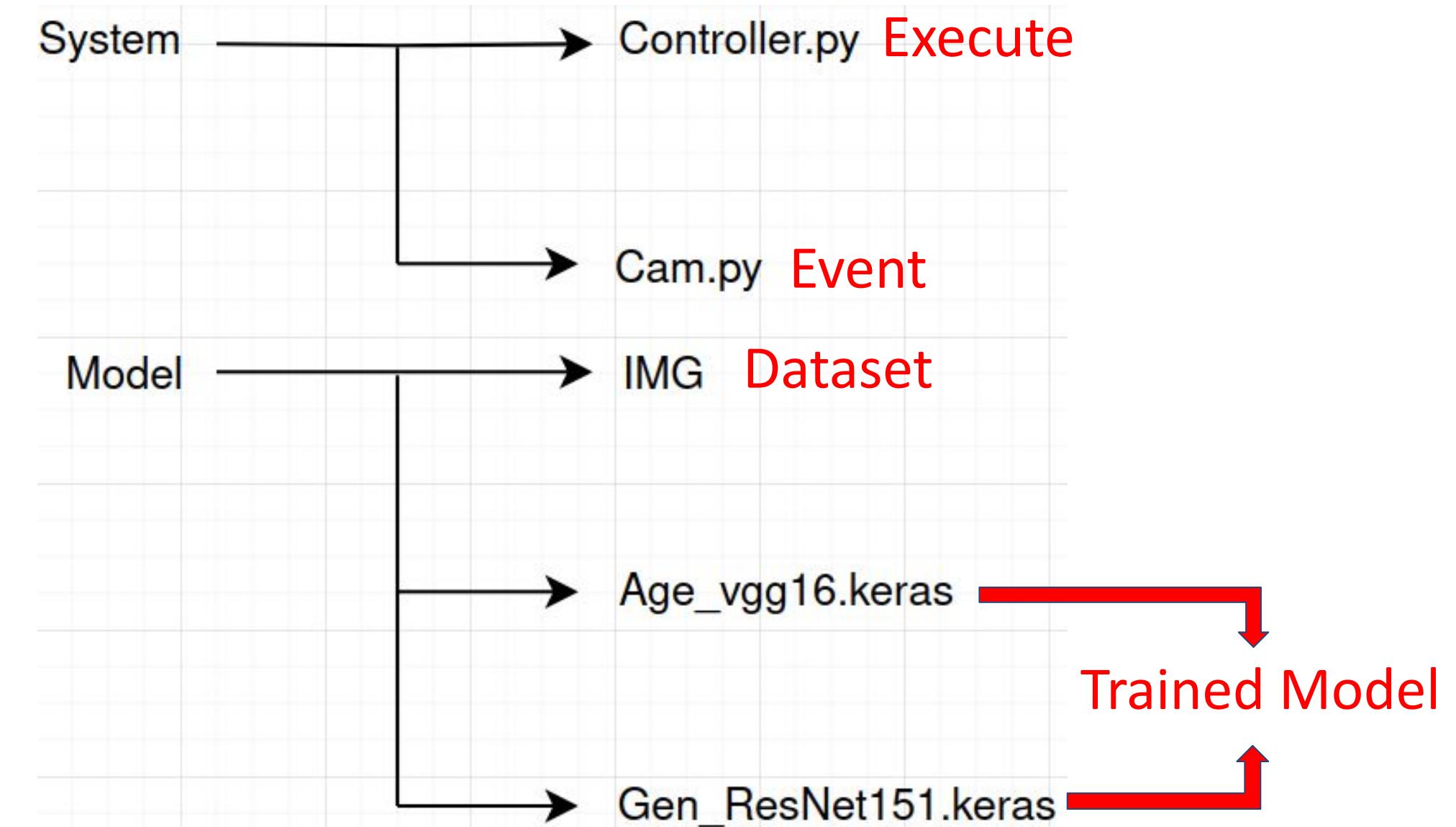
→ 로컬 PC에서 다양한 대규모 언어 모델(LLM)을 쉽게 실행할 수 있도록 돋는 오픈소스 도구.

## USE ollama:1b

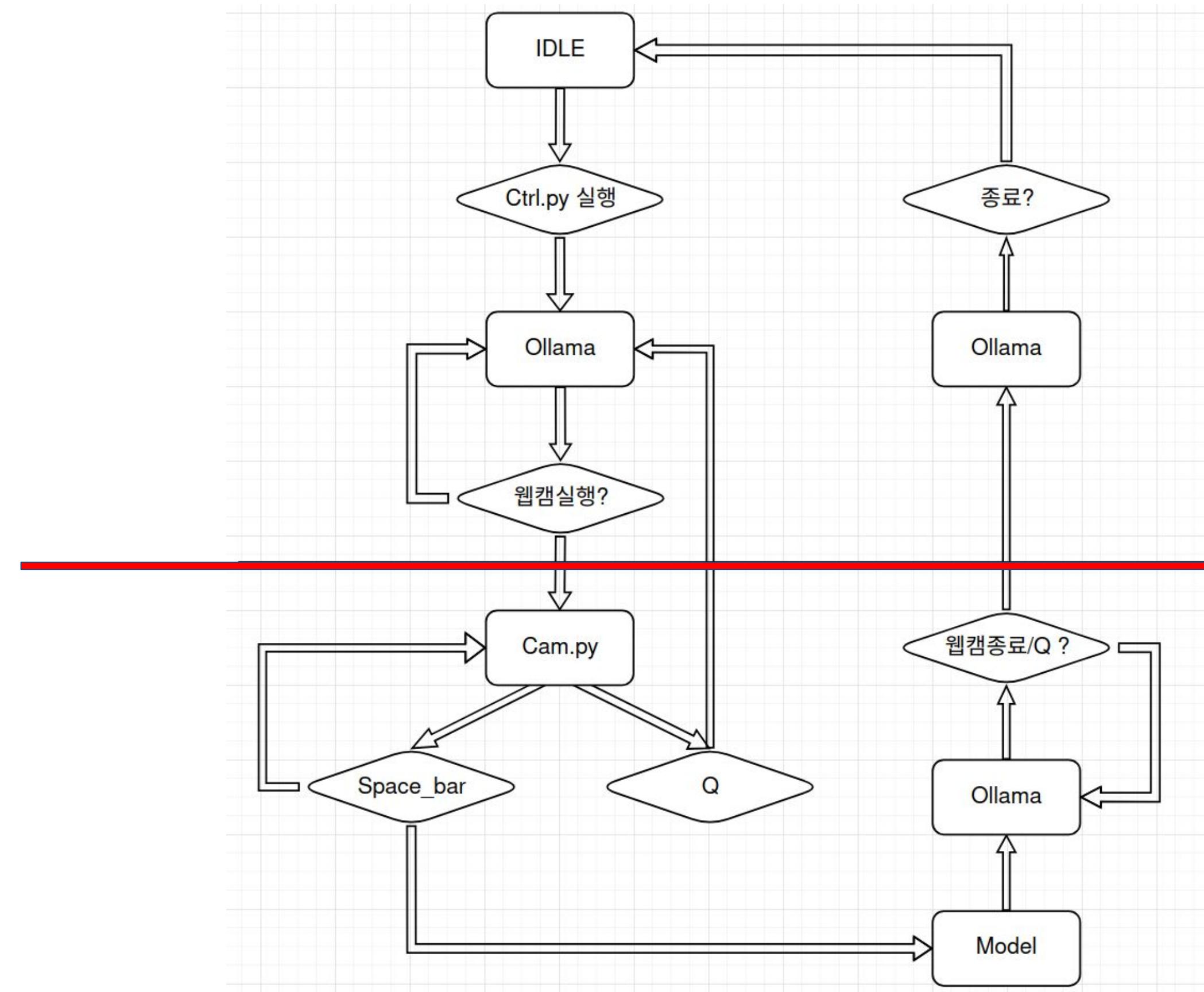
→ 라즈베리파이처럼 리소스가 제한된 환경에서도 구동 가능한 경량 모델에 주로 사용

→ 모델 크기가 작아 메모리와 연산 부담이 적고, 추론 속도도 상대적으로 빠름

# S/W Architecture



# Flow Chart



**Ollama Env**

**CAM Env**

# Main Features

Vgg-16, ResNet-151모델 사용하여 웹캠으로 사람의 나이, 성별을 거의 정확히 추정

```
MX~/final >python ctrl.py  
  
OLLama 컨트롤러 시작!  
OLLama에게 물어보다가 '웹캠실행' 하면 카메라 ON  
'웹캠종료' 하면 카메라 OFF  
'q' 또는 '종료' 입력 시 프로그램 종료  
  
나> 웹캠실행
```



# Main Features

추론한 나이와 성별 값은 자동 완성 프롬프트를 통해 Ollama로 전달 / 답변 하는 동안에는 웹캠 닫음.

[CAM] [LLM 요청] 생성 요청: 나는 34 살 Female 성별의 한국인이야. 필요한 주요 영양제 3가지를 알려주고, 각 영양제를 권장하는 이유를 5줄로 설명해 줘. 답변은 한국어로만 해줘.

[CAM] \*\*2. 오메가-3 지방산\*\*

[CAM]

[CAM] \* \*\*추천 이유:\*\* 오메가-3 지방산은 심혈관 건강 개선, 뇌 기능 향상, 염증 완화 등에 효과적입니다. 여성호르몬 수치 변화는 혈액 내 염증을 증가시키는데 기여하므로, 오메가-3 지방산 섭취는 혈관 건강을 개선하고 스트레스 완화에 도움을 줄 수 있습니다. 또한, 임신 및 출산 중 여성의 건강에도 중요한 영향을 미칩니다.

[CAM]

[CAM] \*\*3. 단백질\*\*

[CAM]

[CAM] \* \*\*추천 이유:\*\* 여성의 신체는 성장, 유지, 재생 과정에서 단백질이 가장 많이 필요합니다. 34세 여성은 나이가 들면서 단백질 섭취량이 감소하기 쉬우므로, 충분한 단백질 섭취는 근육 유지 및 강화, 면역력 증진, 체중 관리 등에 필수적입니다.

[CAM]

[CAM] \*\*추가적으로 고려할 사항:\*\*

[CAM]

[CAM] \* \*\*개인 맞춤 영양:\*\* 위에 추천한 영양제 외에도 개인의 건강 상태, 생활 습관, 목표에 따라 다른 영양제나 보충제를 고려해야 합니다.

[CAM] \* \*\*균형 잡힌 식단:\*\* 영양제는 보조적인 역할이며, 건강한 식단과 규칙적인 운동을 통해 전체적인 건강을 유지하는 것이 가장 중요합니다.

[CAM]

[CAM] \*\*주의:\*\* 영양제 복용 전에는 반드시 전문가와 상담하시기 바랍니다.

[CAM]

[CAM] 웹캠 재개.



# Main Features

---

```
[CAM] [INFO] 이미 추천을 생성 중입니다. 잠시만 기다려 주세요!
[CAM] [INFO] 이미 추천을 생성 중입니다. 잠시만 기다려 주세요!
[CAM] [INFO] 이미 추천을 생성 중입니다. 잠시만 기다려 주세요!
```

답변 준비 중 스페이스 바 입력 예외 처리

```
■ 웹캠 종료 중...
● 웹캠 종료 완료.
■ 실행 중인 웹캠이 없습니다.
나> ■
```

Ollama에 웹캠 종료 명령

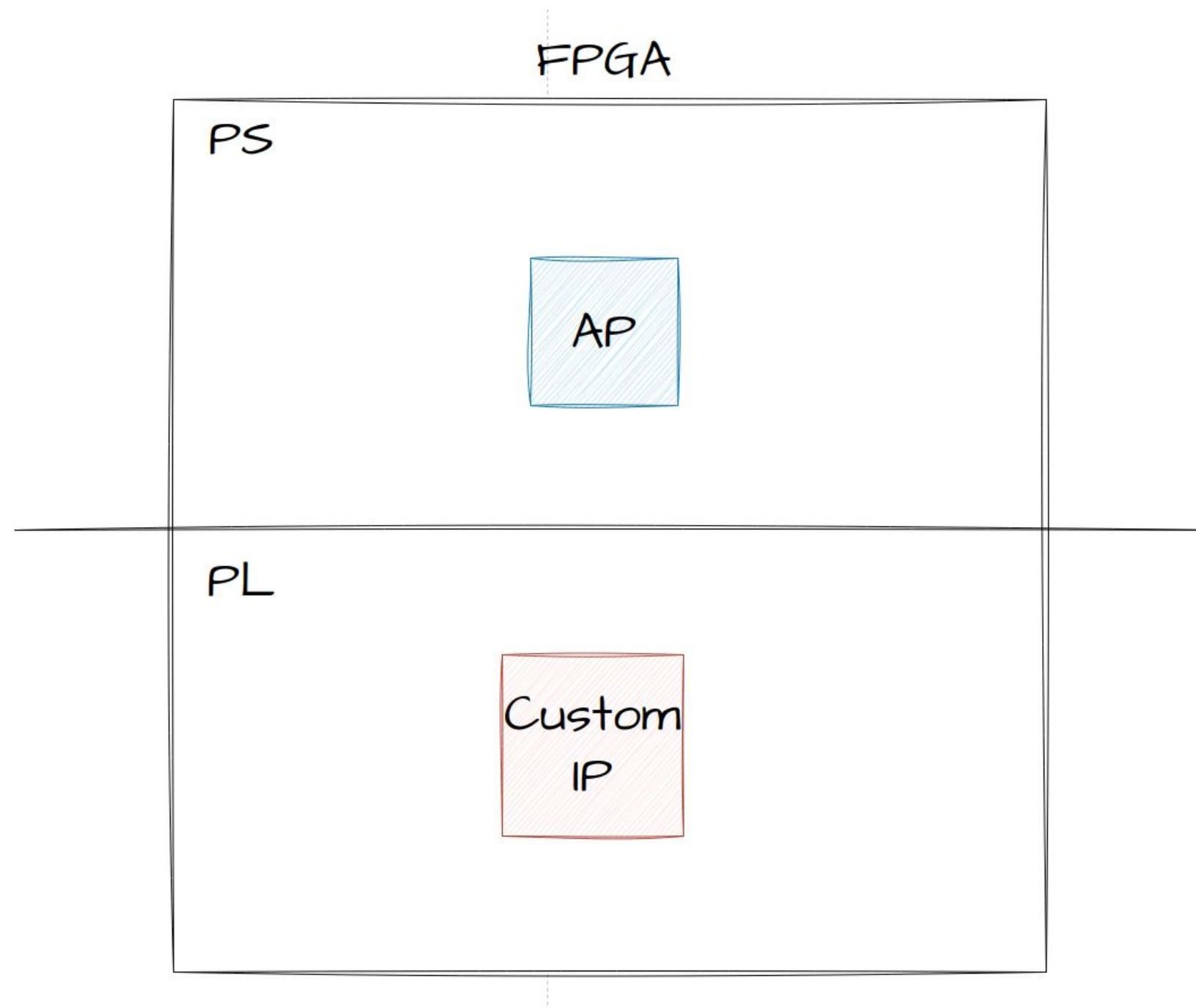
```
나> 종료
■ 실행 중인 웹캠이 없습니다.
컨트롤러 종료.
```

시스템 종료 명령



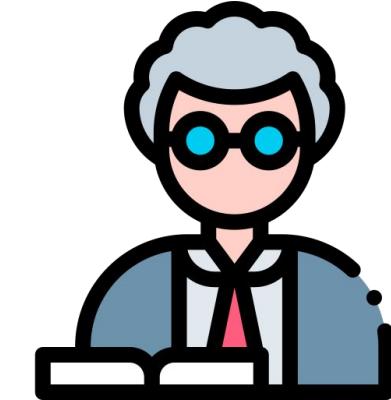
## 04 | 가속기 w FPGA

# AI Accelerator: DPU



## PS

- Processor
- ARM-Core(ARM Cortex - A53)
- 순차 연산 특화

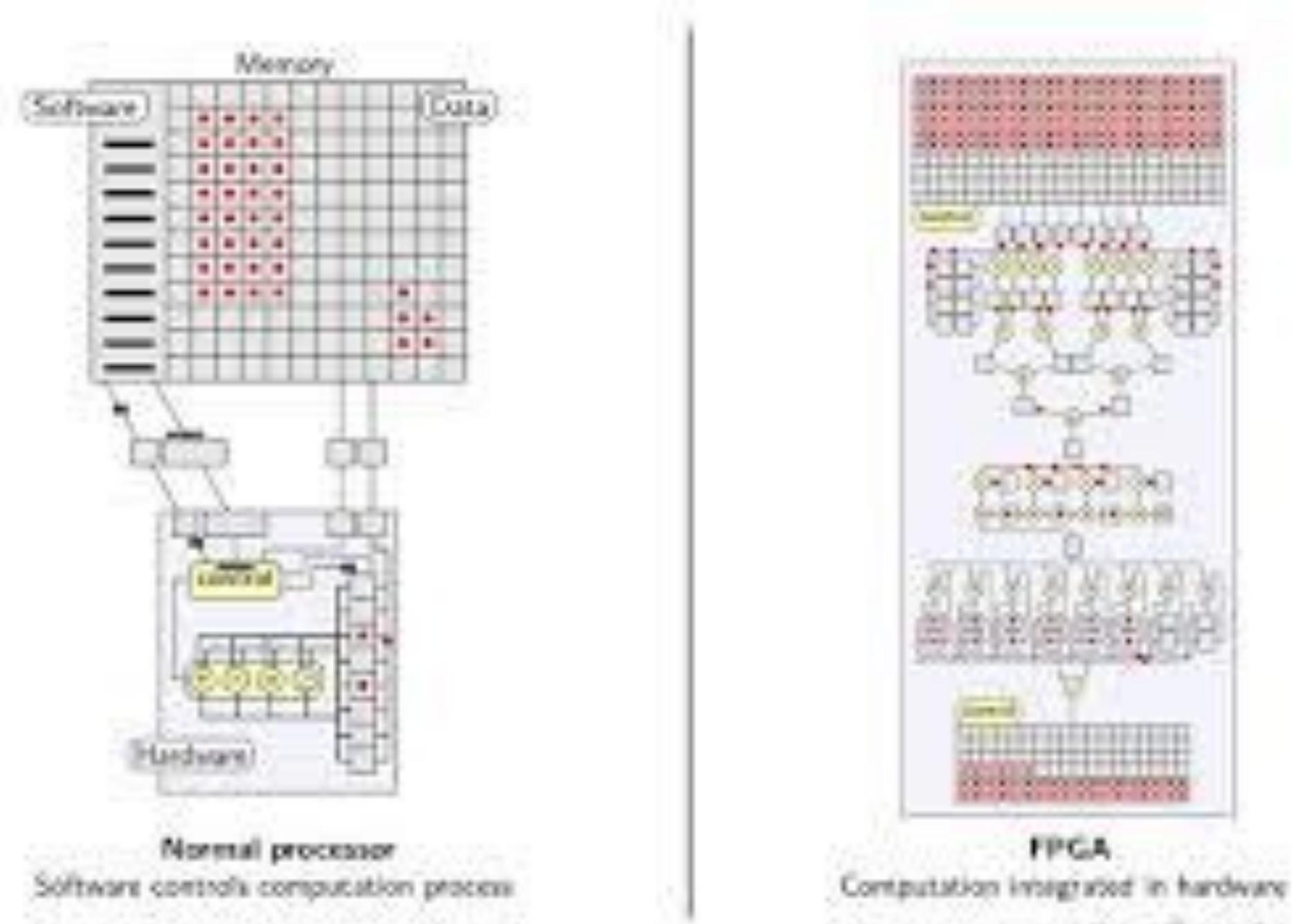


## PL

- Programmable Logic
- 병렬 데이터 처리 특화
- 특정 알고리즘 연산 가속



# AI Accelerator: DPU



<https://www.youtube.com/watch?v=BML1YHZpx2o>

# How To?



## Vivado & Vitis-AI Platform 사용

> HW Level부터 설계

### Pros)

- HW level부터 설계 → 모델별 최적화 가능
- 특정 알고리즘을 위한 가속기 설계 가능

### Cons)

- 설계 난이도 급증
- FPGA linux 이미지를 직접 제작  
→ 검증 X



## PYNQ OS

> 미리 구성된 HW 제공

### Pros)

- 가속기(DPU)가 포함된 리눅스 이미지 제공
- SW 상위수준에서 HW 제어 가능
- 다양한 python 라이브러리 제공

### Cons)

- HW Level에서 최적화 불가능
- Python 라이브러리의 경우, 시스템 OS에 의해 보호됨  
→ 사용자 Customizing에 제약

# BenchMark: Performance Test

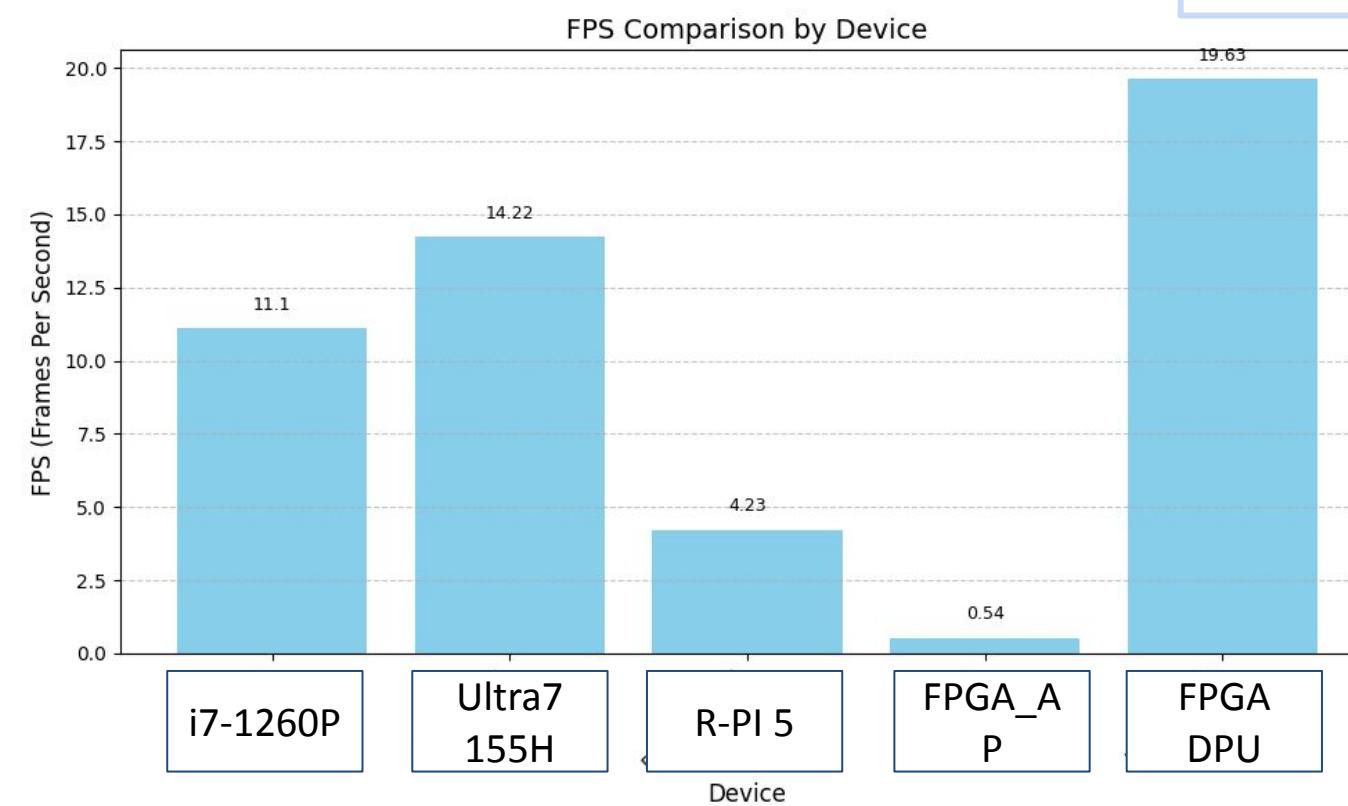


	i7-1260P	Ultra7 155H	RaspberryPi 5 (ARM Cortex A76)	Ultra96V2 (ARM Cortex A53)
Max CLK Speed	4.7GHz	4.8GHz	2.4GHz	1.5GHz

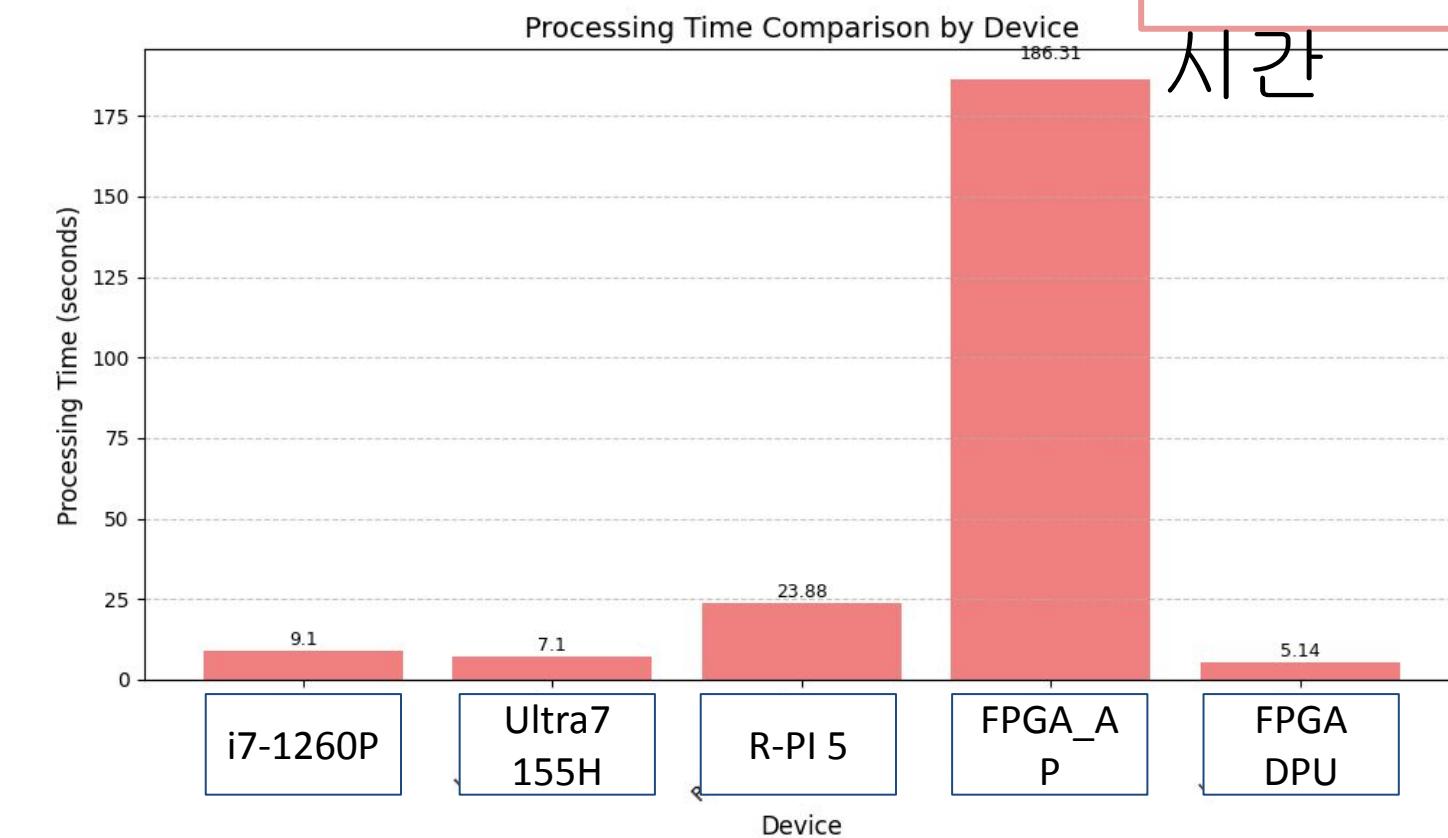
05

# ResNET50

FPS



처리 시간



	i7-1260P	Ultra7 155H	RaspberryPi 5	Ultra96_AP	Ultra96_DPU
FPS	11.1	14.22	4.23	0.54	19.63
처리 시간(sec)	9.10	7.1	23.88	186.31	5.14

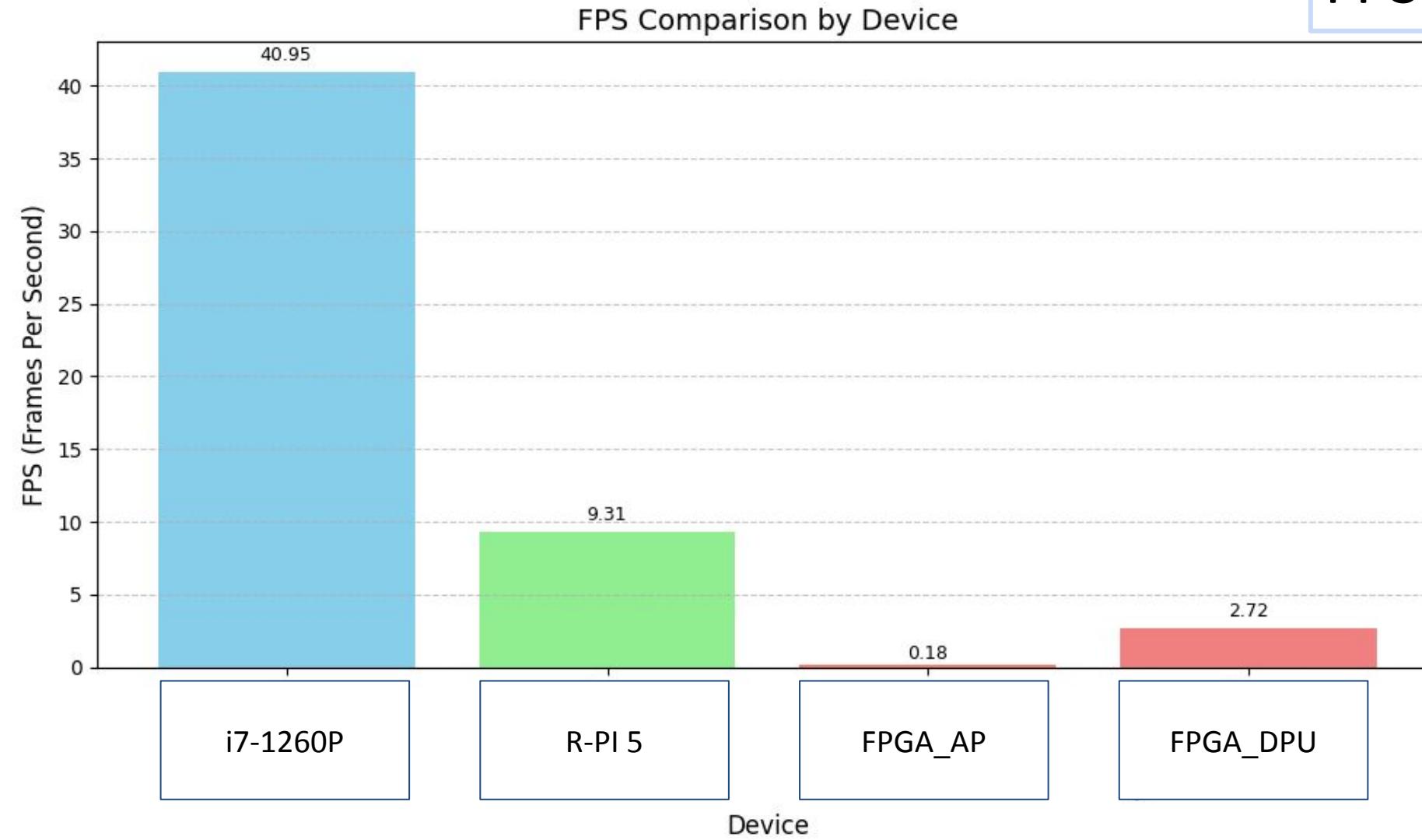


Performance X36 !!

06

# YoloV3

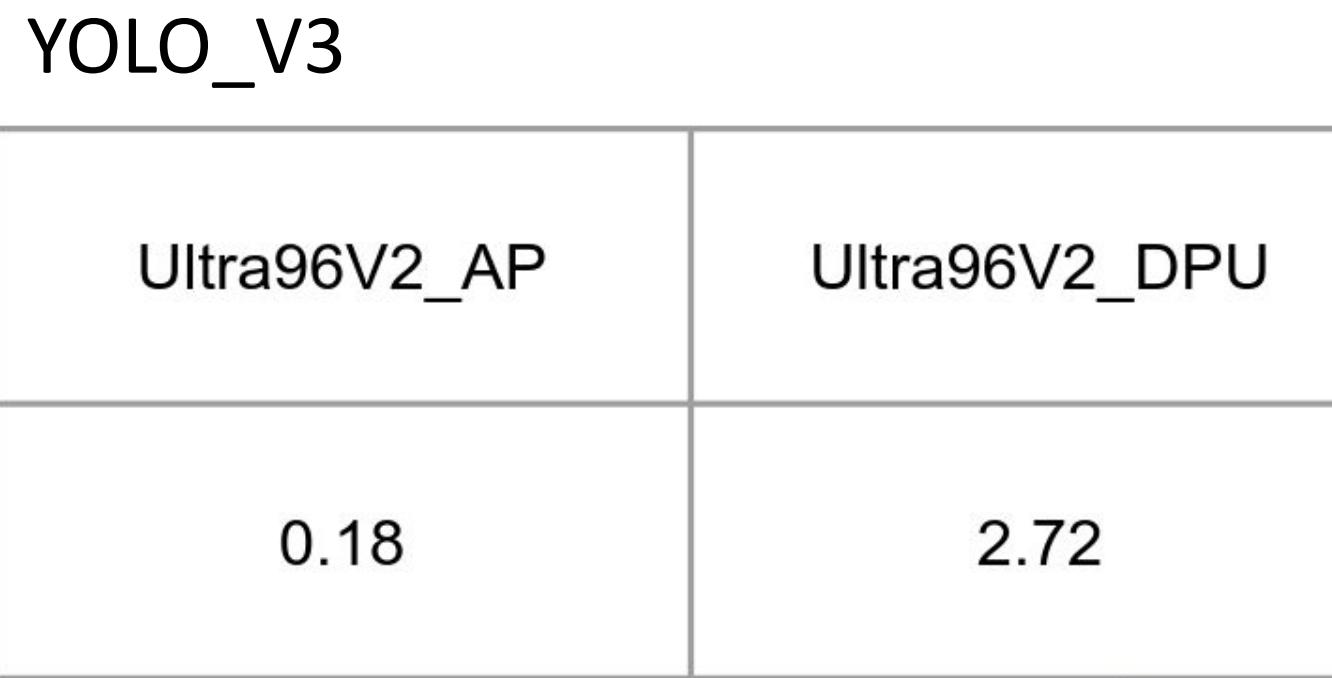
FPS



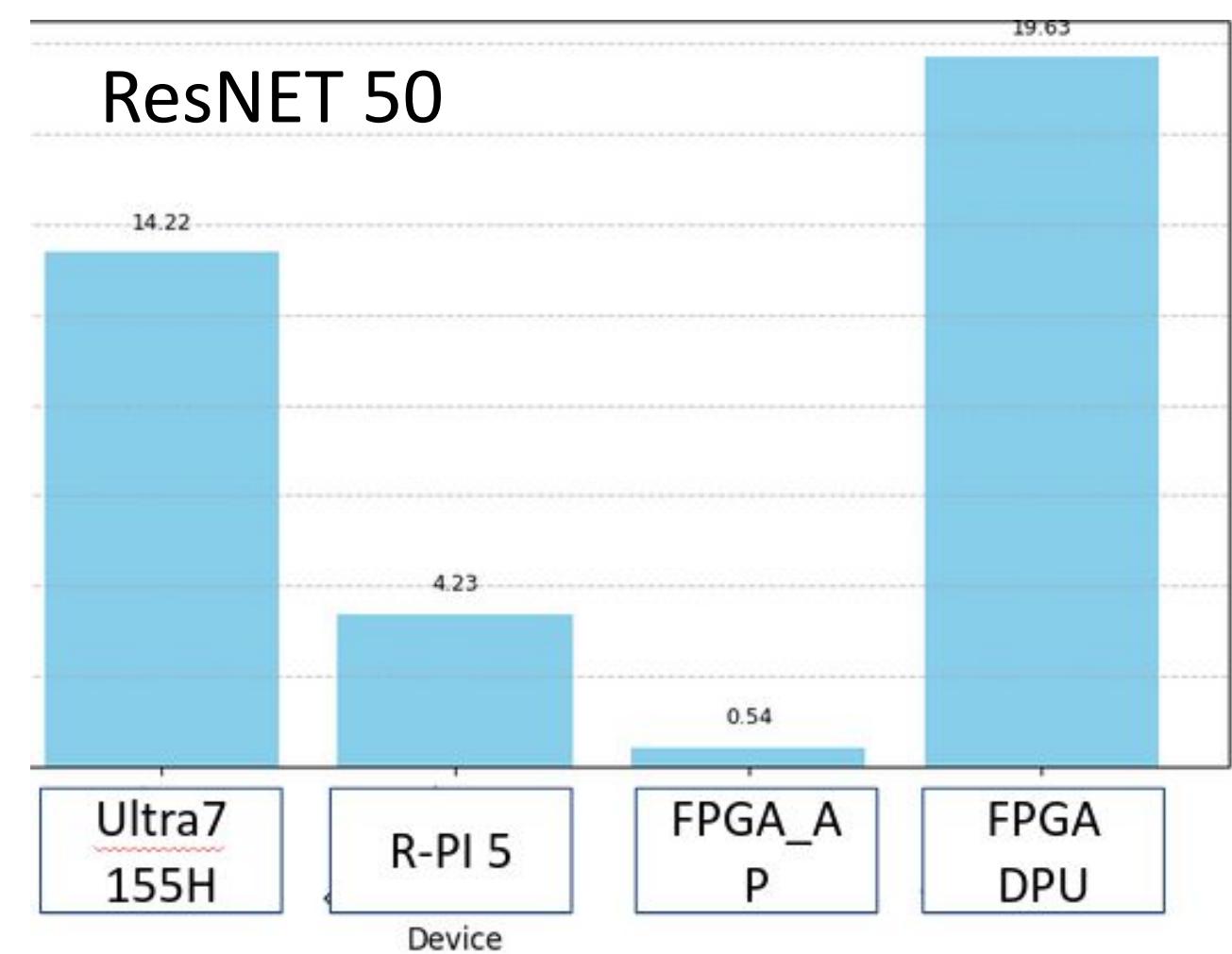
	i7-1260P	RaspberryPi - 5	Ultra96V2_AP	Ultra96V2_DPU
FPS	40.95	9.31	0.18	2.72

Performance X15 !!

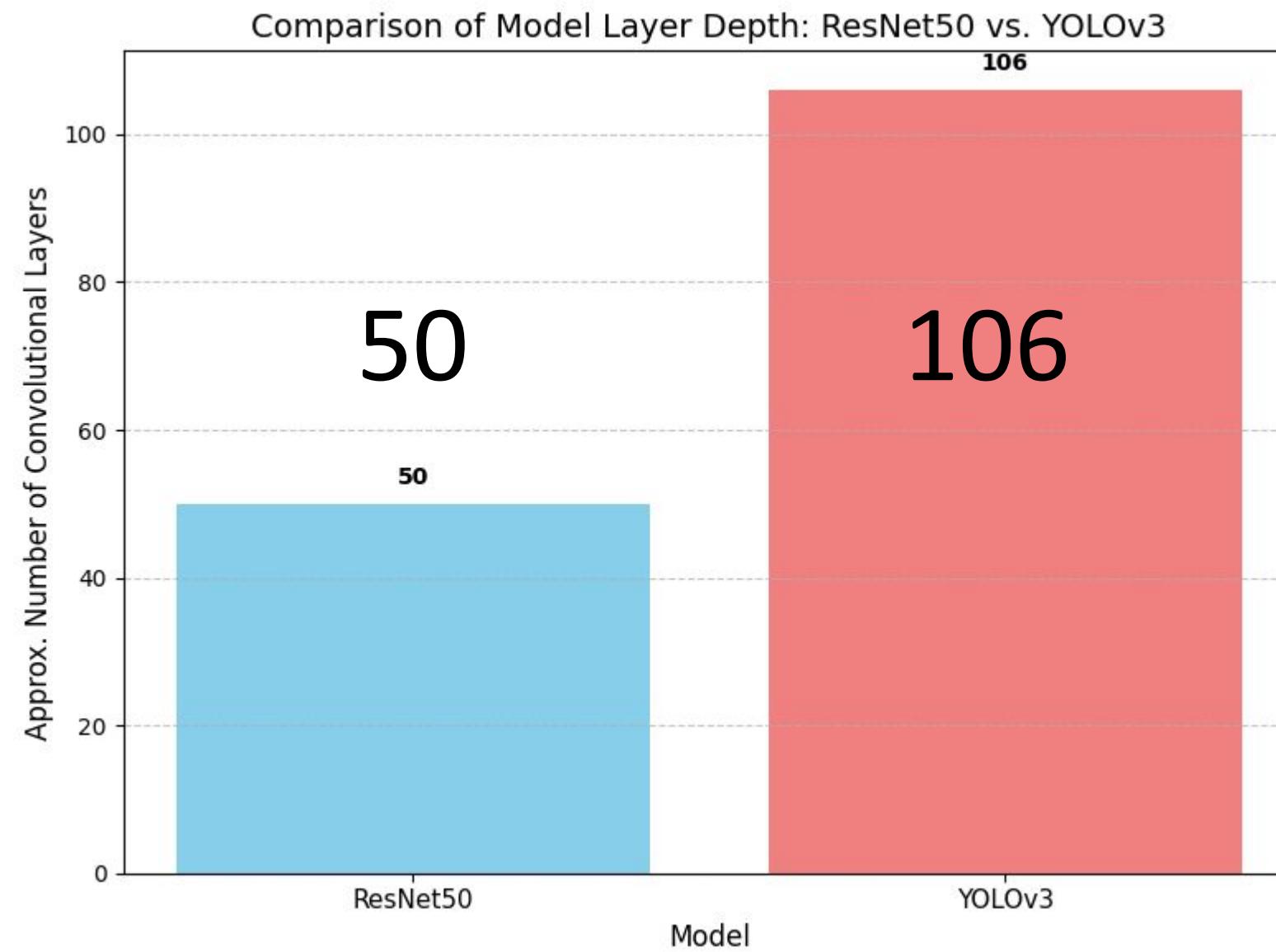
# DPU: ResNet Outperforms YOLO



**Performance X15 !!**



# DPU: ResNet Outperforms YOLO



## Layer 구성

- **ResNET50:** 50 Conv Layer
- **YoloV3:** 53 BackBone(Conv) Layer + 추가 탐지 레이어
  - 후처리 로직(특히 NMS)
  - 순차 처리 + 조건 분기 → CPU가 효율적

## DPU

- 합성곱 연산에 최적화
- 모든 딥러닝 연산에 가속 불가능



## 05 | Trouble Shooting

# 05 Trouble Shooting: Trained AI Model



## Problem

### [DataSet 불균형 문제]

나이 분포가 정규분포가 아닌 비대칭적인 구조  
특히, 20대 데이터가 전체의 대부분을 차지  
→모델이 특정 연령대에 과도하게 최적화되는  
편향(Bias)을 유발 가능성 UP:  
실제로 28살만 출력됨

### [라벨링 오류]

다른 연령대 데이터가 잘못 포함됨



## Solution

### [DataSet 교체 ⇒ 중년층 Data 확보 ]

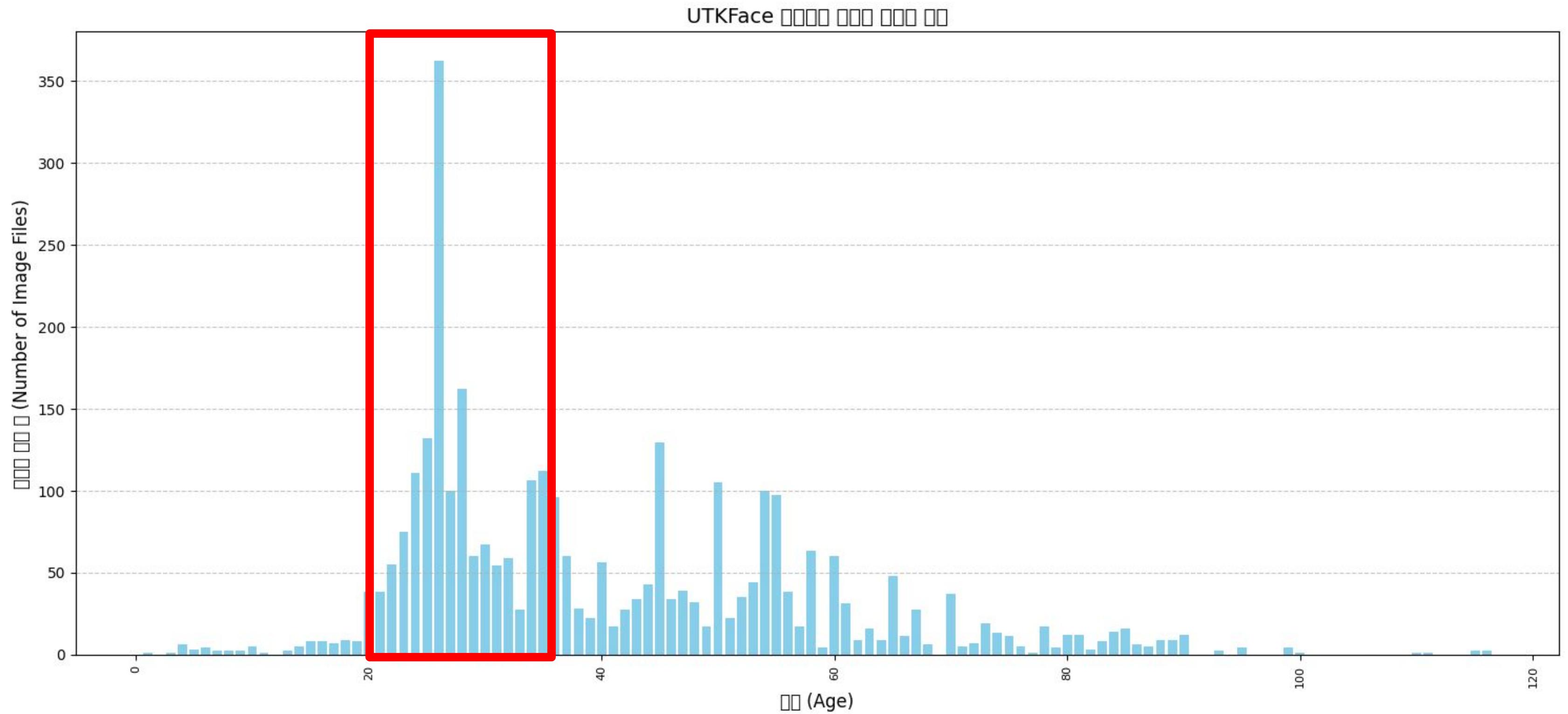
### [학습 모델 변경: 코드 교체]

전이학습:  
Feature Extraction or Fine-tuning 가능 (선택)  
→ Feature Extraction only

### Backbone Model:

<EfficientNetB3> →  
<VGG16, ResNet50V2/152V2, Xception,  
InceptionV3, MobileNetV3Small/Large>

# 05 Trouble Shooting: Trained AI Model



# 05 Trouble Shooting: Trained AI Model



## Problem

### [낮은 학습 성능]

**WebCam** 추론 진행 시,  
실시간으로 나이 추정값이 변화 → 속도 ▼  
나이 추정 변동성이 큼 → 정확도 ▼

### Image

추론 진행 시,  
WebCam 대비 → 정확도가 크게 ▼  
얼굴 인식을 못 하는 문제 ▲  
→ 얼굴 각도, 크기, 10년 전 데이터에 기반한  
얼굴



## Solution

### [학습 데이터 교체 및 개수 변화]

20,000(D1) → 900(D2) → 10,000(D2) →  
2,000(D3)

얼굴을 인식하고 나이를 추정할 충분한 시간 제공  
(3s)  
후 고정 값 출력 → 정확도 및 속도 개선

**Age:** 데이터 품질 즉, 나이 분포도 → 성능

**Gender:** 학습 데이터 개수 → 성능

### [추론 모델 후보정 처리]

UTKFace Data가 아닌 일반 사진으로 평가  
→ 경향성을 평가 및 후보정에 반영

# 05 Trouble Shooting : Integrated System



## Problem

웹캠 프레임에서 얼굴 검출과 추론을 수행하면서 영상이 버벅이고 느려지는 현상 발생

- 매 프레임마다 얼굴 검출 및 추론
- CPU 과도 사용
- 웹캠 영상 끊기거나 응답 지연 발생



## Solution

예측 빈도를 낮춰 불필요한 추론 연산을 감소시켜 시스템 부하를 감소시킴

- 초당 1번만 추론 되게 진행
- CPU 사용률 감소, 상대적으로 영상 부드럽게 출력
- 정확도의 큰 차이는 없음!

# 05 Trouble Shooting : Integrated System



## Problem

### [Raspberry PI에서 모델 추론 지연]

→ 초당 1회만 추론하도록 제한했음에도 불구하고, 웹캠과 Ollama를 동시에 실행하면 모든 CPU 코어가 사용되어 성능 저하 및 시스템 불안정 문제가 발생함.

-> 최적화 필요!



## Solution

### [리소스 관리 중심의 알고리즘 최적화]

→ Ollama가 답변을 준비하는 동안 웹캠을 닫고, 답변이 완료되면 웹캠을 다시 재개하는 방식

추론 중 발생하던 CPU 과부하와 영상 지연을 해결하고, 웹캠과 LLM 추론을 안정적으로 병행함.

# 05 Trouble Shooting : Accelerator



## Problem

[웹캠 송출 및 OS 이미지 충돌]

평상 시

PYNQ OS의 **Xwindow** 화면 송출

웹캠 구동 시

Xwindow와 웹캠 출력이 **DP Port**에서 충돌



## Solution

[스위칭 옵션 추가]

“print(overlay.ip\_dict)”

→ DP 포트에 해당하는 IP 및 구조 파악

→ DisplayPort라는 Pynq 라이브러리 제어

코드 실행 시: Xwindow OFF/ Webcam ON

코드 종료 시: 원상복구

# 05 Trouble Shooting : Accelerator



## Problem

### [.xmodel 파일 생성]

PYNQ-DPU를 효율적으로 사용 위해선

AI 모델 → xmodel파일 변환 필요

#### [Problem]

1. Vitis-AI 미지원 모델 존재
2. .xmodel 파일 사이즈 大 → 커널 Down



## Solution

### [지원 버전]

Vitis-AI 컨테이너 제공 xmodel 리스트 확인

→ 해당 리스트에 존재하는 모델 이용

#### [경량화]

Keras → tflite 파일 변환 후 xmodel 변환



## 06 | 시연 및 고찰

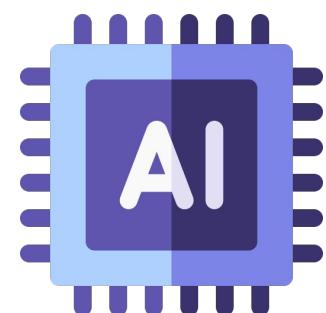
**"웹캠 실행" 입력**

01

# 고찰

병렬 데이터 多  
(Conv)

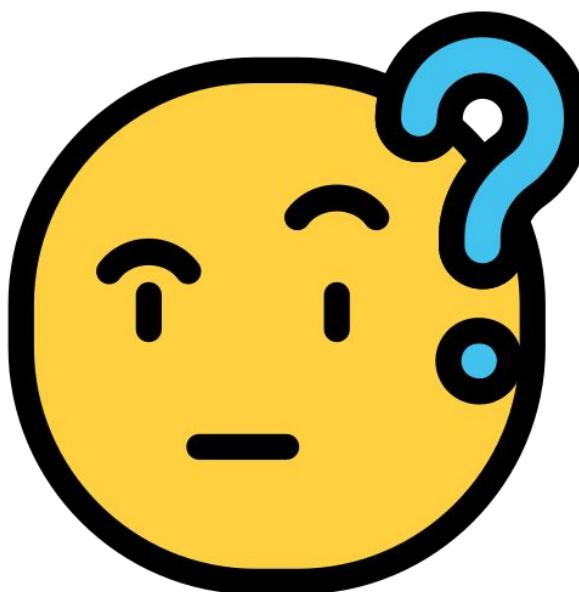
DPU



Performance ↑

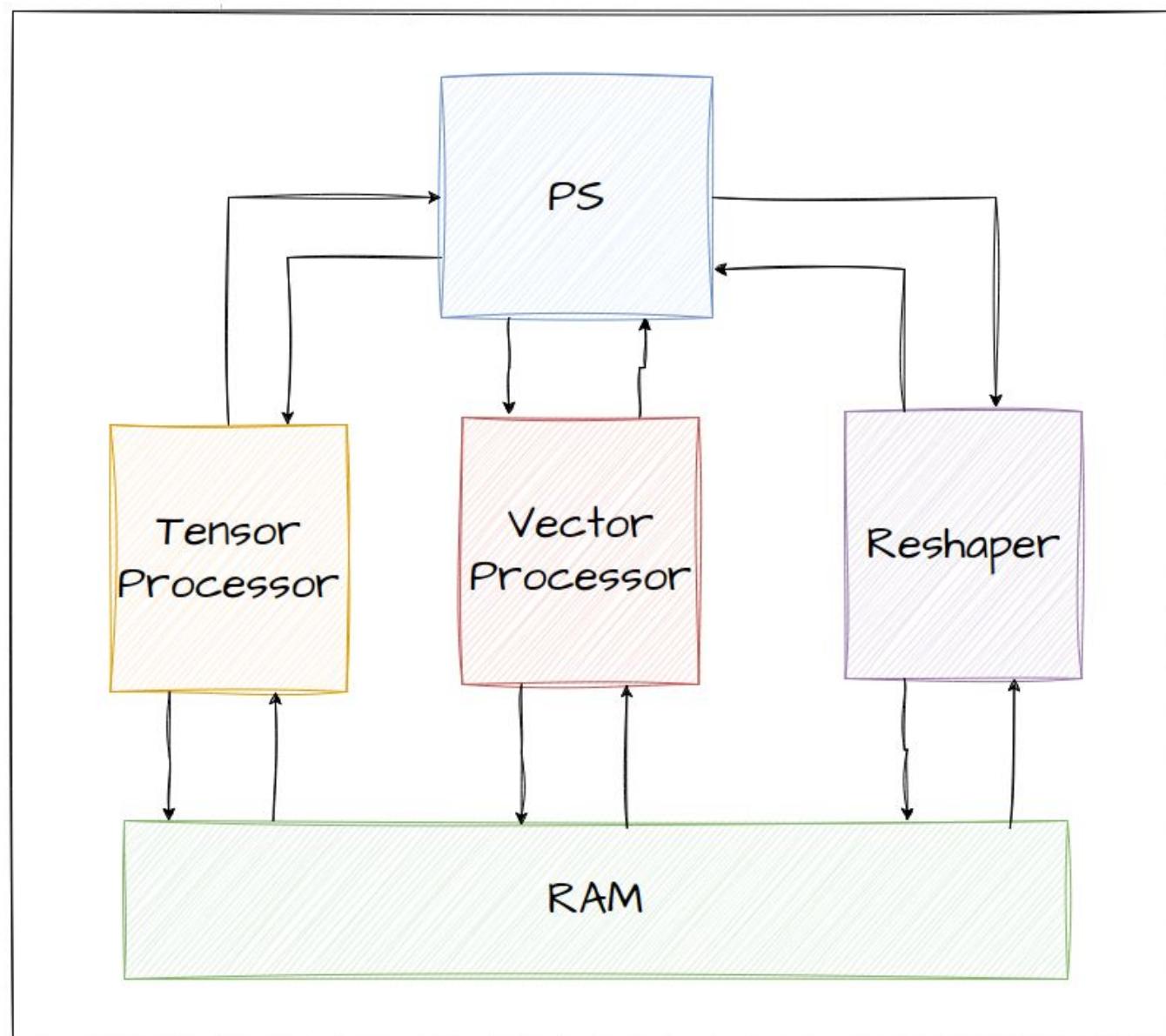
Sometimes  
CPU << PS w DPU

복잡한 후처리  
연산과 단순 요소  
연산은?



합성곱 연산을 제외한 나머지 연산을  
가속하는  
**보조 Processor**

# 고찰



	<b>Tensor Executor</b>	<b>Vector Executor</b>
<b>주요 연산</b>	Matrix Multiplication Convolution	벡터 요소별 연산 활성화 함수 정규화
<b>병렬 처리</b>	대규모 병렬연산 처리 (여러 행렬 곱셈 동시 처리)	SIMD 병렬 연산 (하나의 명령어로 여러 벡터 요소 처리)
<b>연산 분야</b>	신경망 주요 레이어 (Conv, FC)	Pooling Batch Activation Fn
<b>연산 복잡도</b>	대량의 병렬 데이터 연산 특화	간단한 반복작업 특화
<b>주요 목적</b>	AI 모델의 핵심 연산 가속	데이터 전처리/후처리 보조 연산

참고:

<https://velog.io/@hyal/%EB%B2%94%EC%9A%A9%EC%A0%81%EC%9D%B8-NPU-%EA%BC%9C%EB%9C%EA%B8%B0-2nd-1-TOP-Architecture>

## 기대 효과

### 01 | 데이터 I/O 효율성 향상

Tensor Executor가 일하는 동안 Vector Executor가 전/후 처리 연산 수행 → **Pipelining** 성능 향상

### 02 | 범용성 및 유연성 증대

합성곱 혹은 병렬 연산에 특화된 Tensor를 Vector가 보조해줌 → **다양한 딥러닝 모델**을 가속 가능

### 03 | 전력 효율성 향상

기존: 병렬 데이터 처리 이외 연산 CPU 부담 → Vector Executor 도입으로 **CPU 부하를 감소**

# Q&A