Computer Vision

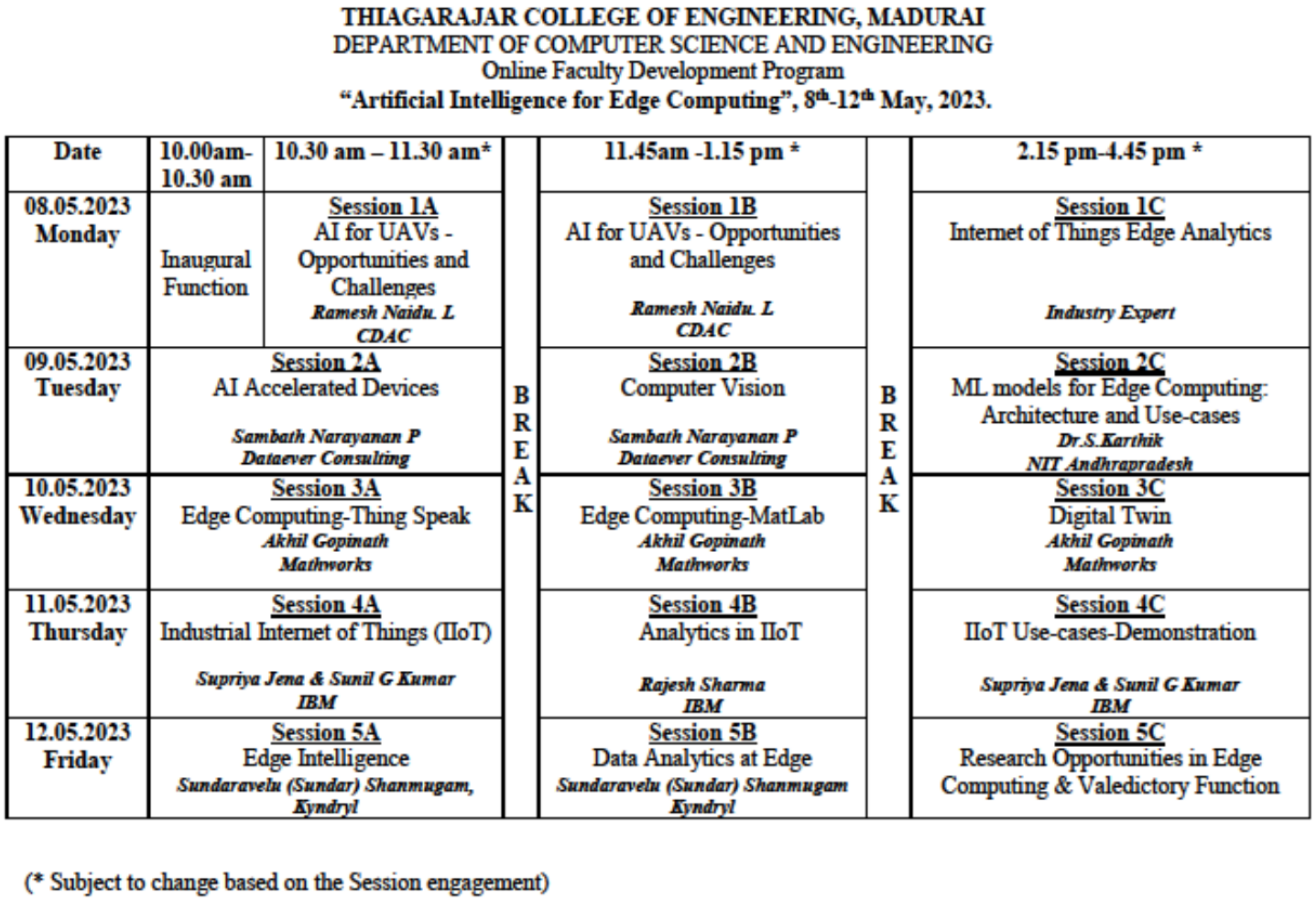
9th May 2023,Day2, Session 2B

sambath.narayanan@gmail.com

Acknowledgement

Thankyou !

- In this presentation we have used many tools, libraries and images from NVIDIA. We acknowledge the same.
- We built the HandPose demo in our lab using the youtube video provided by make2Explore, https://youtu.be/FPXVC7WrC4s
- In other places, if we use some information, we provide the URL for the same
- If we missed out something, we will correct the same



THIAGARAJAR COLLEGE OF ENGINEERING, MADURAI
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
Online Faculty Development Program
"Artificial Intelligence for Edge Computing", 8th-12th May, 2023.

Date	10.00am-10.30 am	10.30 am – 11.30 am*		11.45am -1.15 pm *		2.15 pm-4.45 pm *
08.05.2023 Monday	Inaugural Function	**Session 1A** AI for UAVs - Opportunities and Challenges ***Ramesh Naidu. L CDAC***	B R E A K	**Session 1B** AI for UAVs - Opportunities and Challenges ***Ramesh Naidu. L CDAC***	B R E A K	**Session 1C** Internet of Things Edge Analytics ***Industry Expert***
09.05.2023 Tuesday	**Session 2A** AI Accelerated Devices ***Sambath Narayanan P Dataever Consulting***			**Session 2B** Computer Vision ***Sambath Narayanan P Dataever Consulting***		**Session 2C** ML models for Edge Computing: Architecture and Use-cases ***Dr.S.Karthik NIT Andhrapradesh***
10.05.2023 Wednesday	**Session 3A** Edge Computing-Thing Speak ***Akhil Gopinath Mathworks***			**Session 3B** Edge Computing-MatLab ***Akhil Gopinath Mathworks***		**Session 3C** Digital Twin ***Akhil Gopinath Mathworks***
11.05.2023 Thursday	**Session 4A** Industrial Internet of Things (IIoT) ***Supriya Jena & Sunil G Kumar IBM***			**Session 4B** Analytics in IIoT ***Rajesh Sharma IBM***		**Session 4C** IIoT Use-cases-Demonstration ***Supriya Jena & Sunil G Kumar IBM***
12.05.2023 Friday	**Session 5A** Edge Intelligence ***Sundaravelu (Sundar) Shanmugam, Kyndryl***			**Session 5B** Data Analytics at Edge ***Sundaravelu (Sundar) Shanmugam Kyndryl***		**Session 5C** Research Opportunities in Edge Computing & Valedictory Function

(* Subject to change based on the Session engagement)

Overall Session Agenda

Computer Vision(CV)

- Why Computer Vision
- What is Computer Vision
- CV Frameworks of importance
- Data sets
- Computer Vision Models
- State of the art in Computer Vision

CV applications /use cases

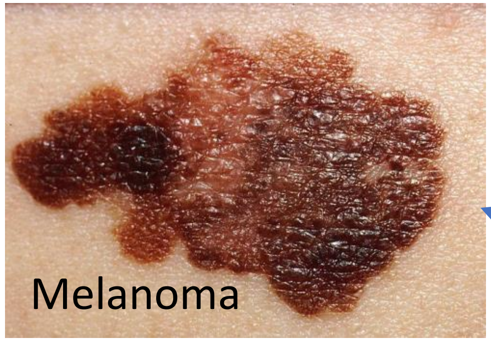
- Manufacturing
- Automotive
- Healthcare
- Surveillance

CV Demos

- Basic CV Demos
- Industrial - Welding / Segmentation
- Healthcare - Covid
- Video Surveillance - Car burglary

Why Computer Vision(CV)?

CNN DL Model vs. Skin specialist



abnormal growth or appearance compared to the skin around it

“ These findings show that deep learning CNN are capable of outperforming dermatologists, in the task of detecting melanomas "

- Holger Haenssle, MD, senior managing physician, University of Heidelberg, Germany

Skin specialist

- Having additional information improved the doctor‘s performance where the mean specificity 75.7%, and the mean ROC AUC 0.82.

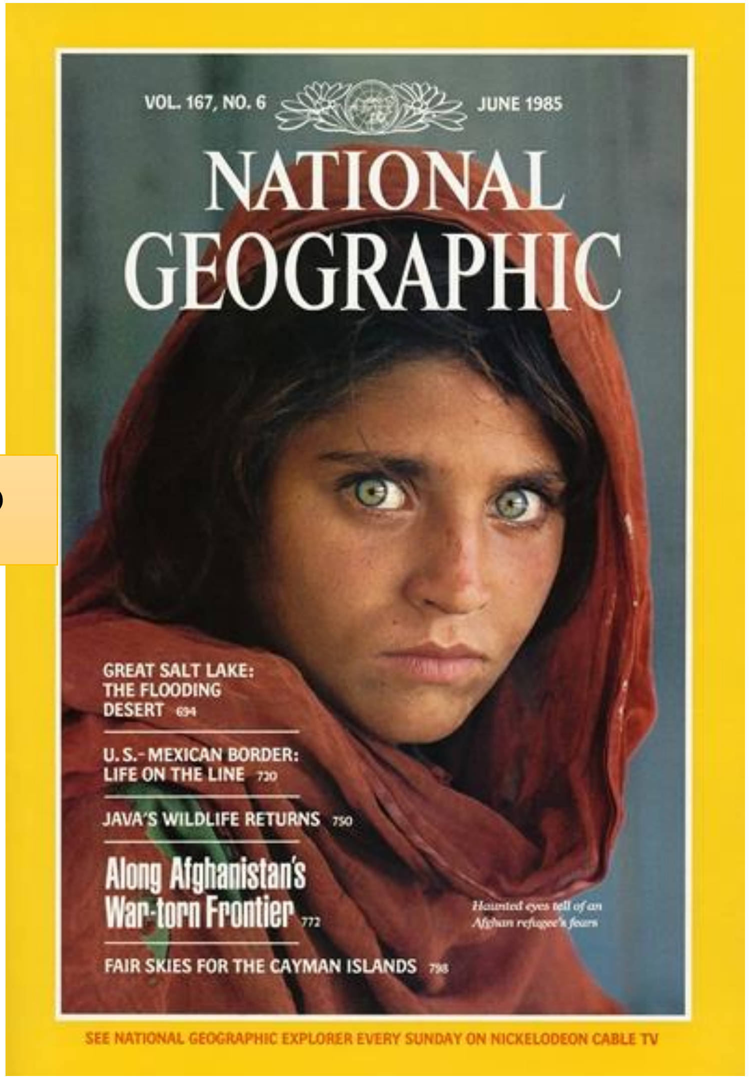
CNN performance

- on the same reader study level-II test showed , a specificity of 90%, and an ROC AUC of 0.95.

Annals of Oncology, May 2020

ROC - Receiver Operating Characteristic

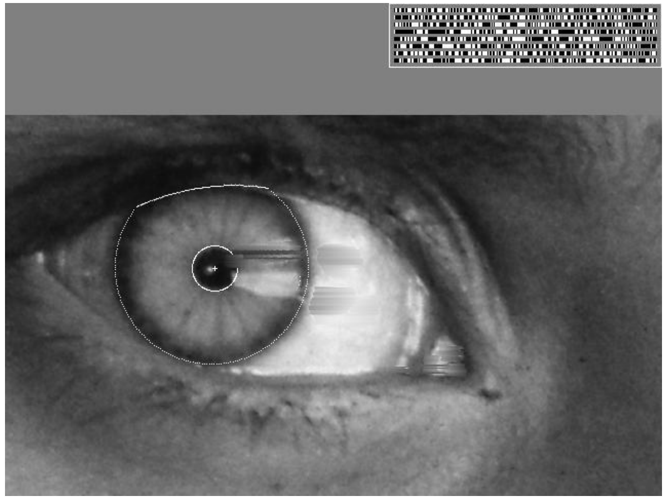
How the Afghan Girl was Identified by Her Iris Patterns

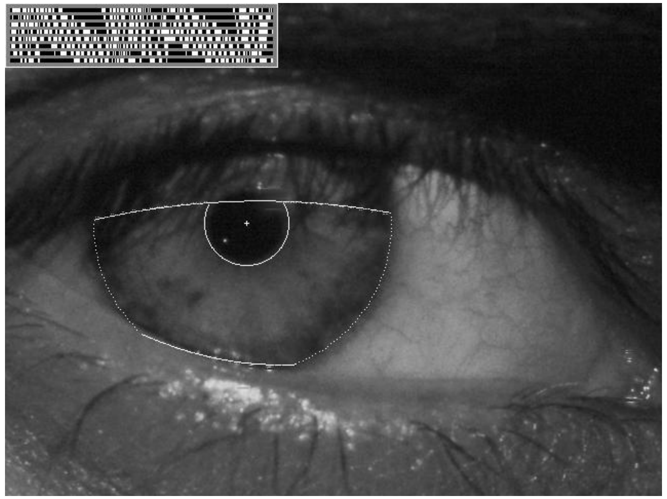

VOL. 167, NO. 6
JUNE 1985
NATIONAL GEOGRAPHIC
GREAT SALT LAKE: THE FLOODING DESERT 694
U.S.-MEXICAN BORDER: LIFE ON THE LINE 720
JAVA'S WILDLIFE RETURNS 750
Along Afghanistan's War-torn Frontier 772
Haunted eyes tell of an Afghan refugee's fears
FAIR SKIES FOR THE CAYMAN ISLANDS 798
SEE NATIONAL GEOGRAPHIC EXPLORER EVERY SUNDAY ON NICKELODEON CABLE TV

Who is she?

Vision-based biometrics

“How the Afghan Girl was Identified by Her Iris Patterns”

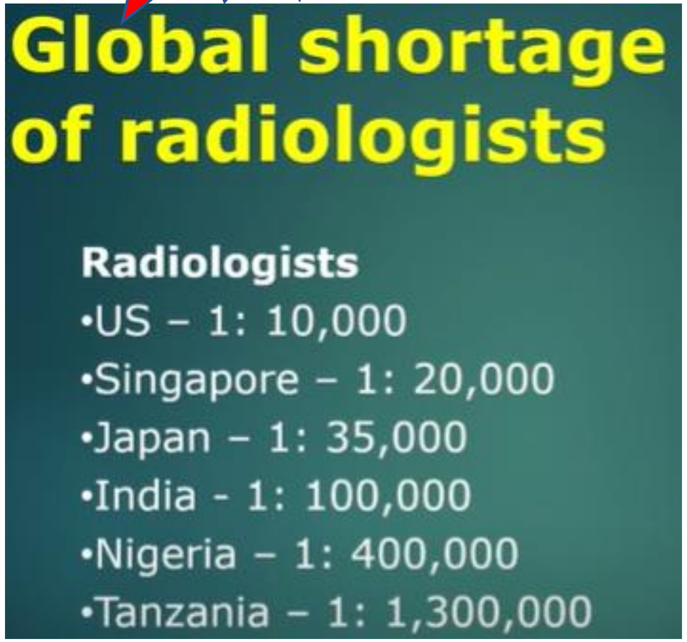




Can AI address Healthcare industry Challenges ?

- **Radiologist** is a medical practitioner trained in analyzing MRI or CT scan, X Ray and using the analysis to diagnose medical conditions

PRE COVID Statistics



Global shortage of radiologists

Radiologists

•US – 1: 10,000
•Singapore – 1: 20,000
•Japan – 1: 35,000
•India - 1: 100,000
•Nigeria – 1: 400,000
•Tanzania – 1: 1,300,000

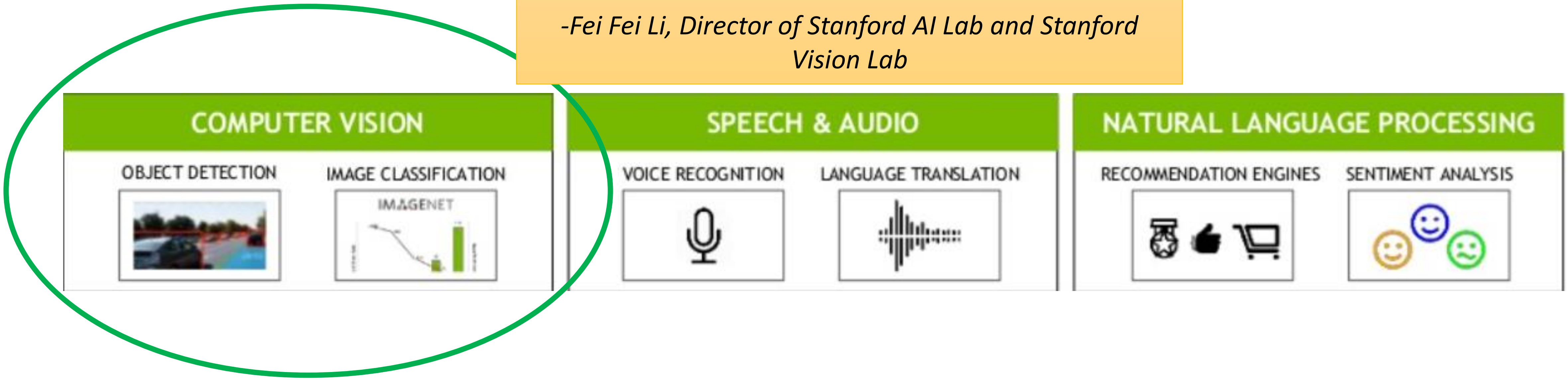
Number of scans professionals have to analyze = 1,000 a day

This can lead to lengthy delays between scan and treatment – **even when someone needs urgent care**.

What is Computer Vision(CV)?

Computer Vision is the most important branch of AI

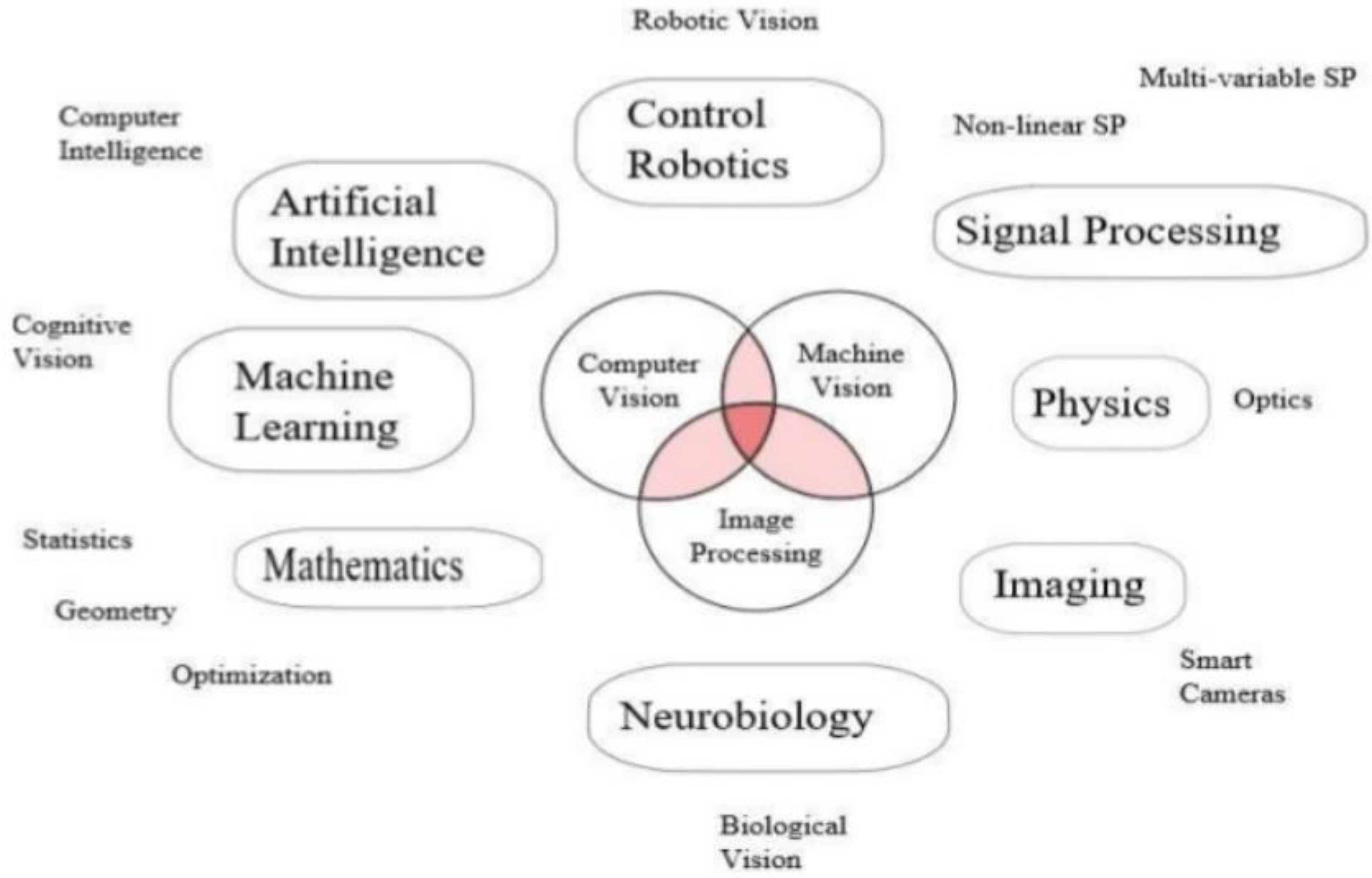
> *"If we want machines to think, we need to teach them to see."*
>
> *-Fei Fei Li, Director of Stanford AI Lab and Stanford Vision Lab*



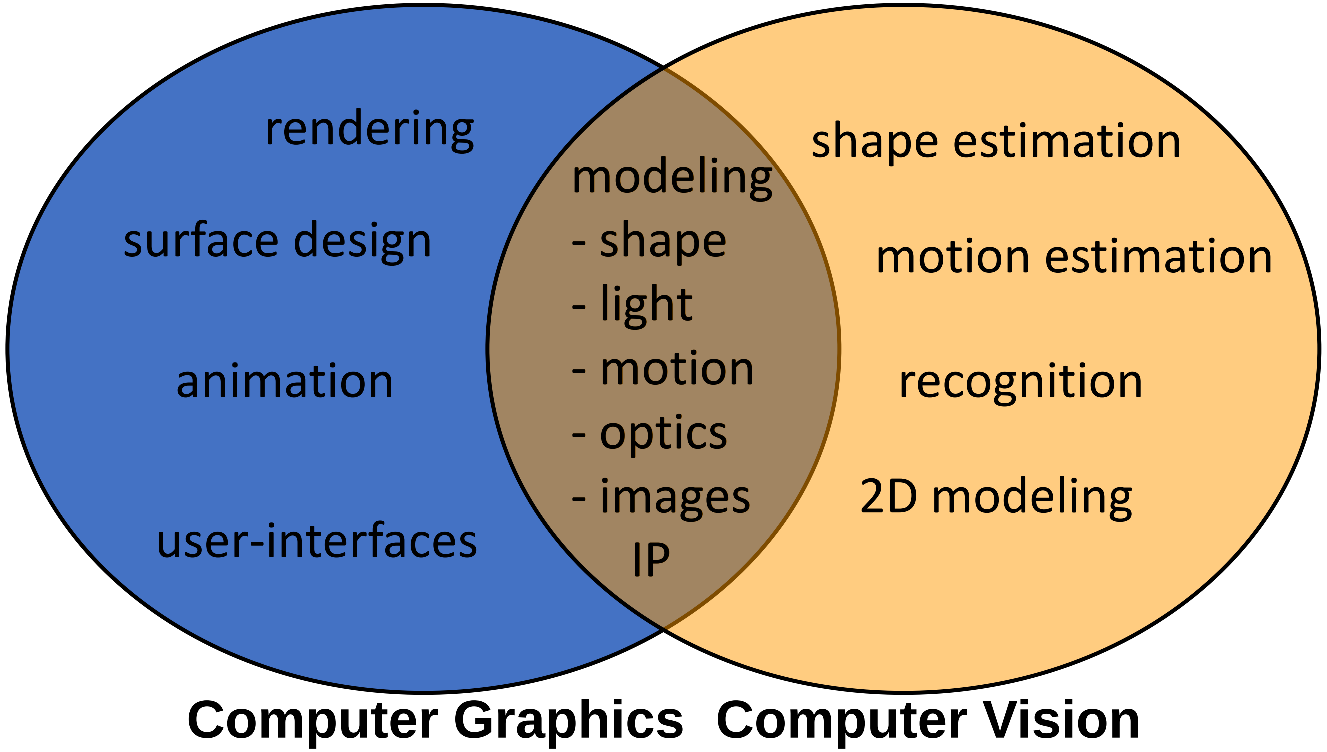
Neural networks and Deep Learning are Making Computer Vision More Capable of Replicating Human Vision

"Machine Vision Fundamentals, How to Make Robots See". *NASA Tech Briefs Magazine*. **35** (6)

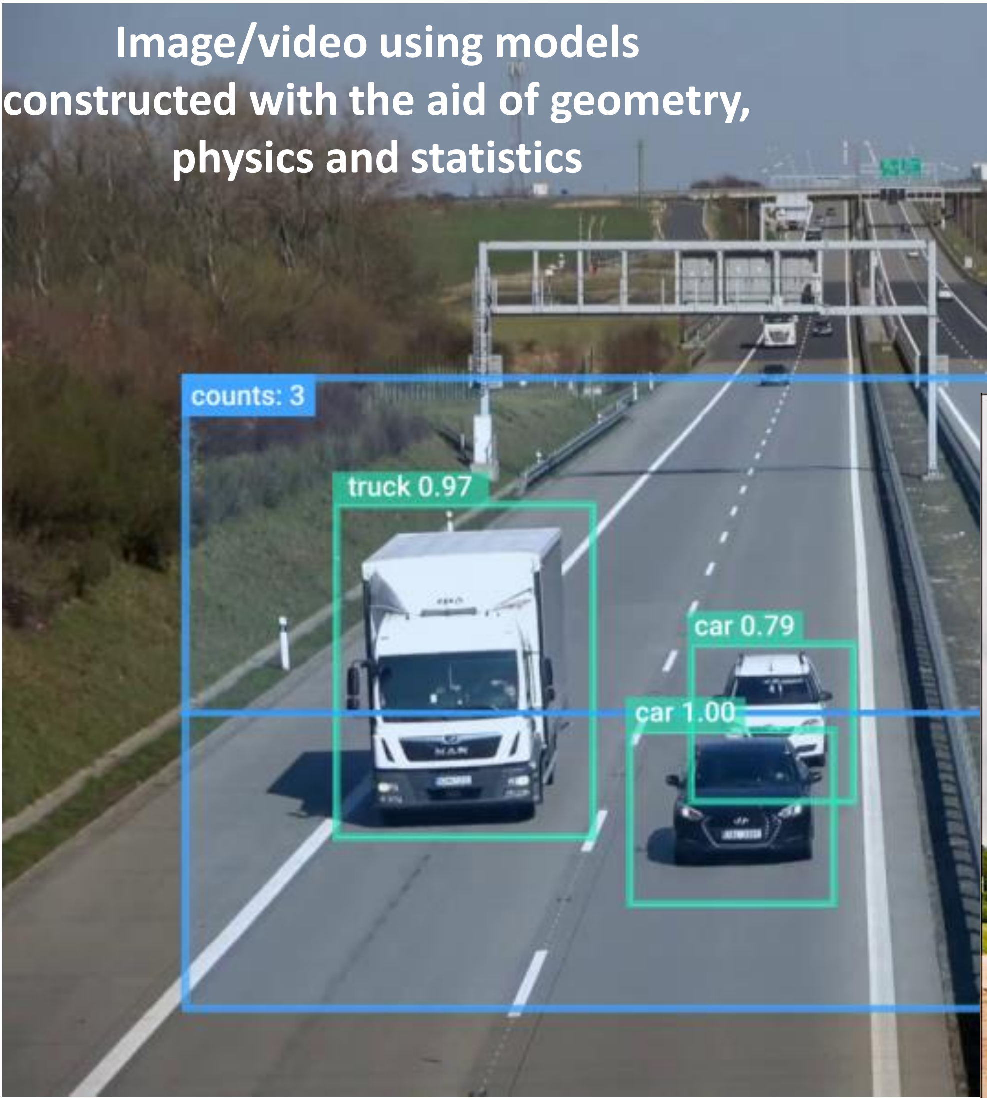
Computer Vision



Intersection of Vision and Graphics

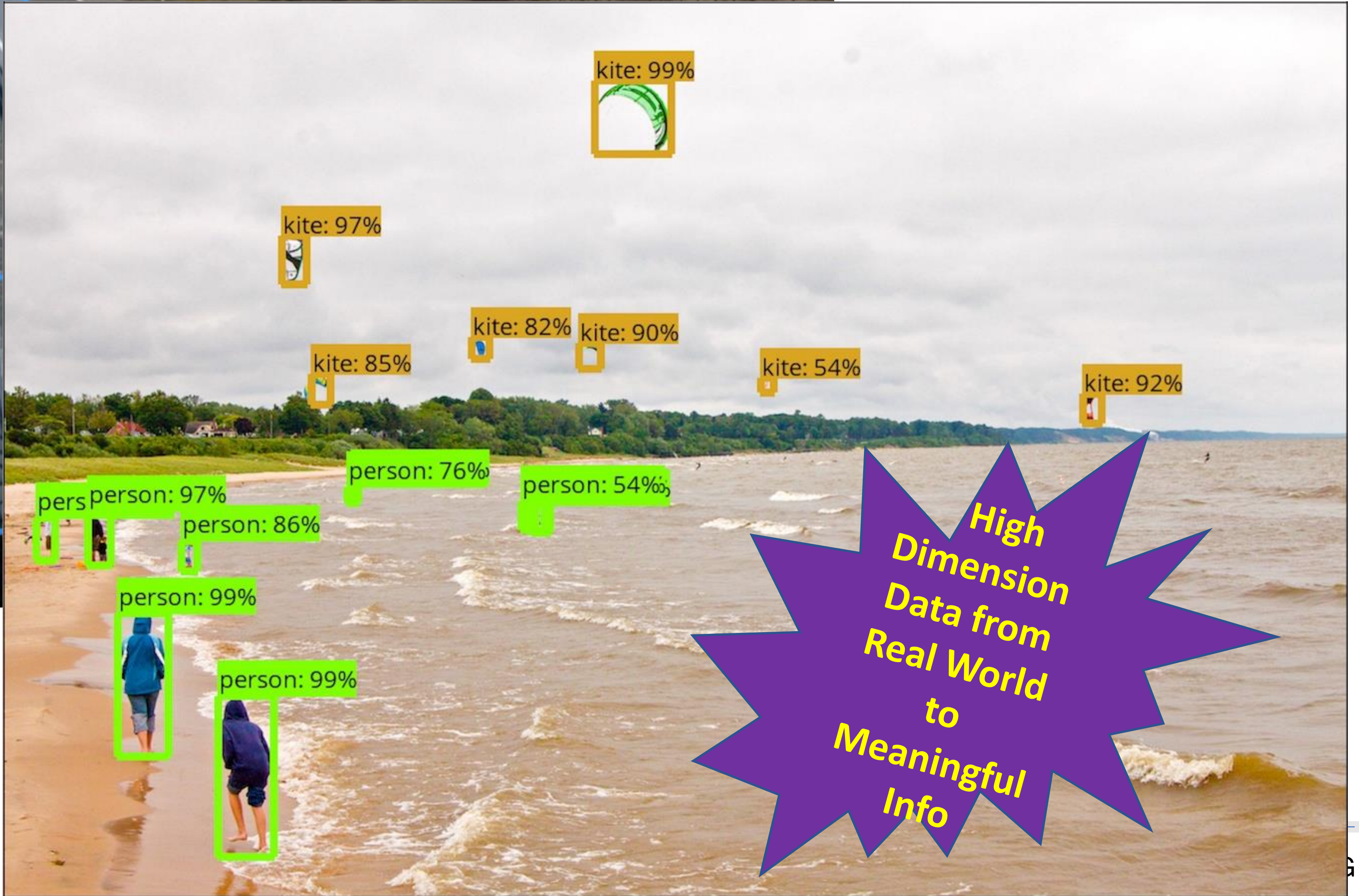


Computer Vision tasks



Image/video using models constructed with the aid of geometry, physics and statistics

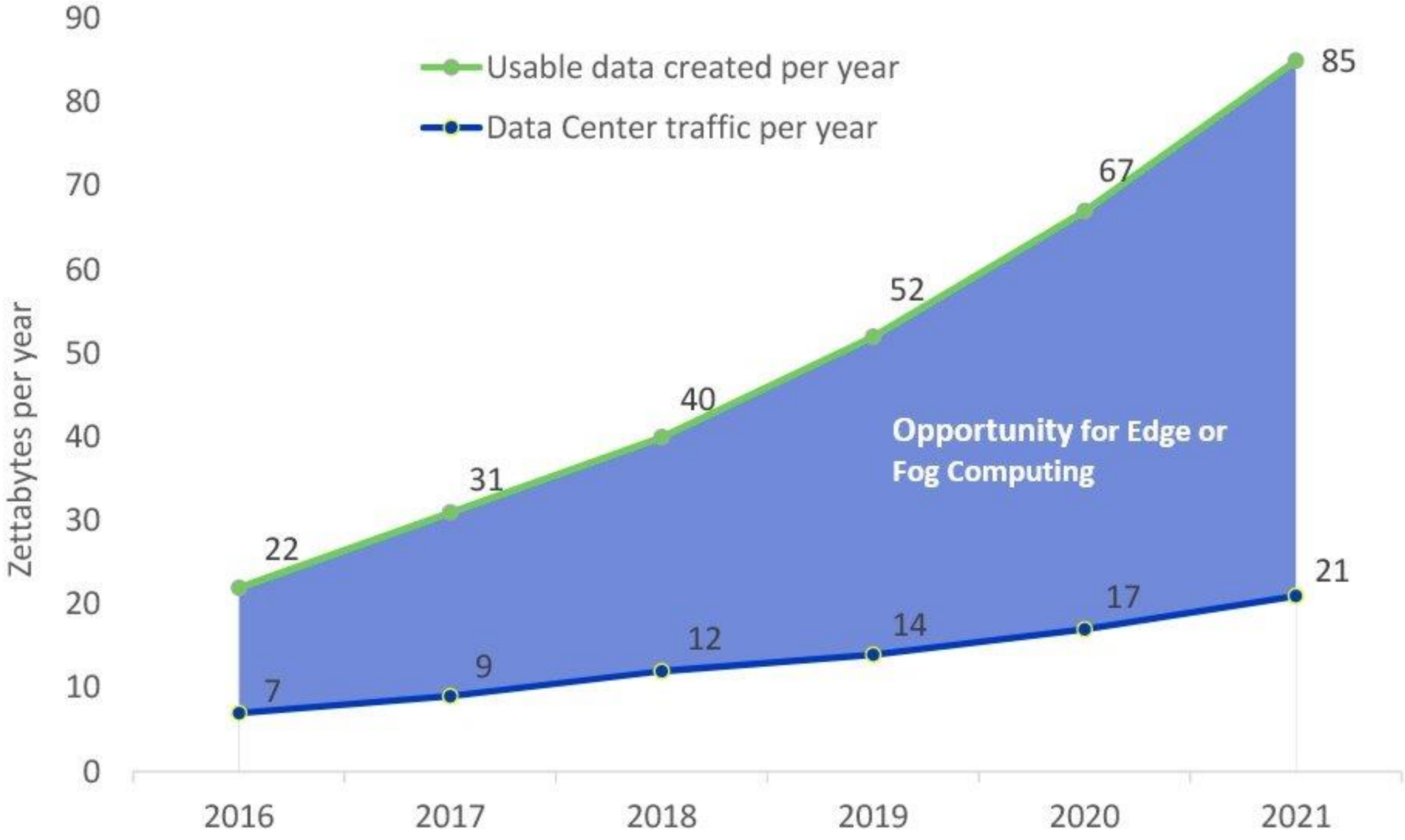
"Machine Vision Fundamentals, How to Make Robots See". *NASA Tech Briefs Magazine*. **35** (6)

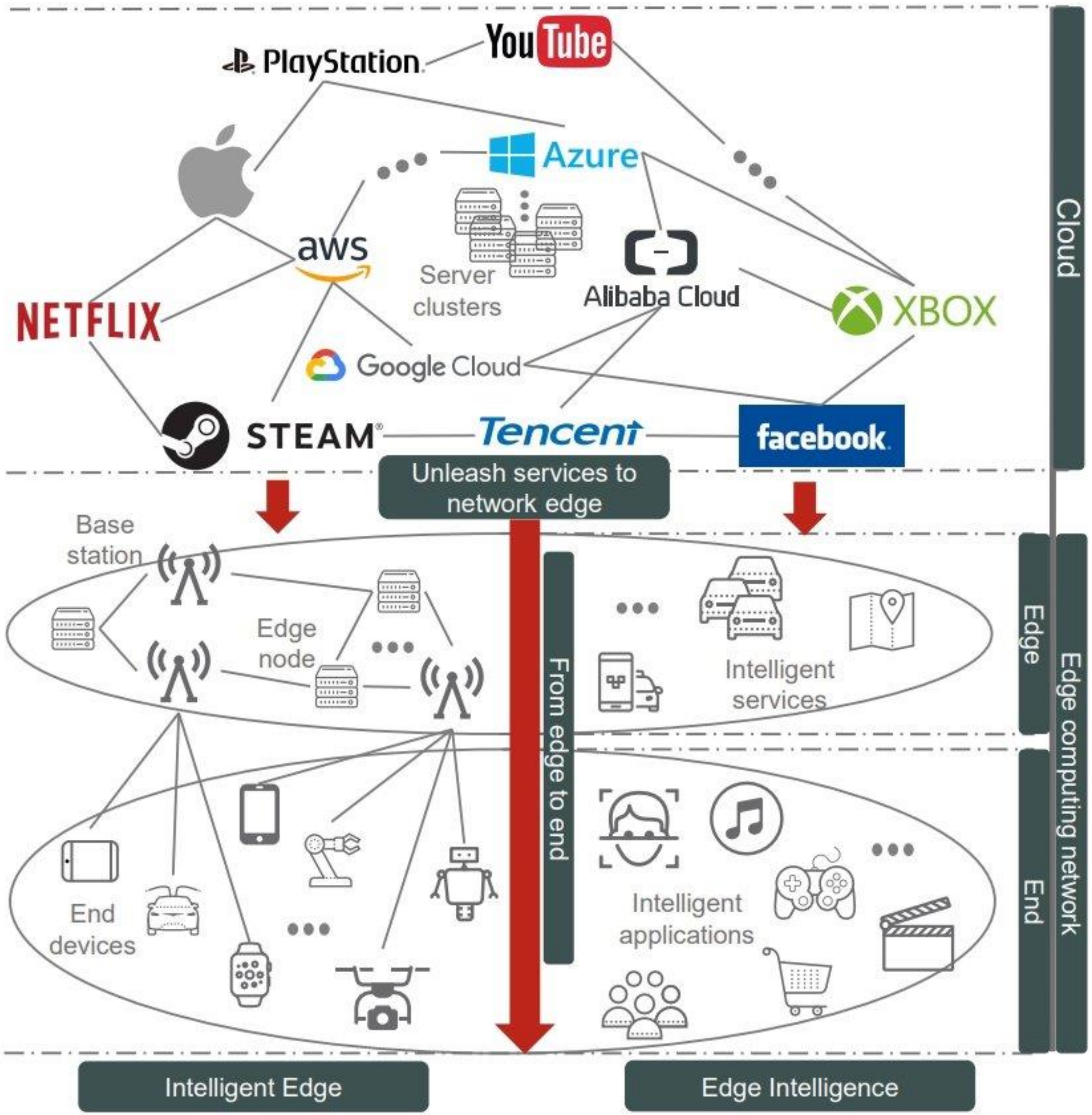


High Dimension Data from Real World to Meaningful Info

Use Cases of Computer Vision in Manufacturing

- Measurement / Gauging
- Guidance, part tracking, and identification
- Optical character recognition and verification (OCR/OCV)
- Barcode / 2D readers and Pattern recognition
- Vision guided Robotics
- Quality Assurance – package / part inspection / defect detection

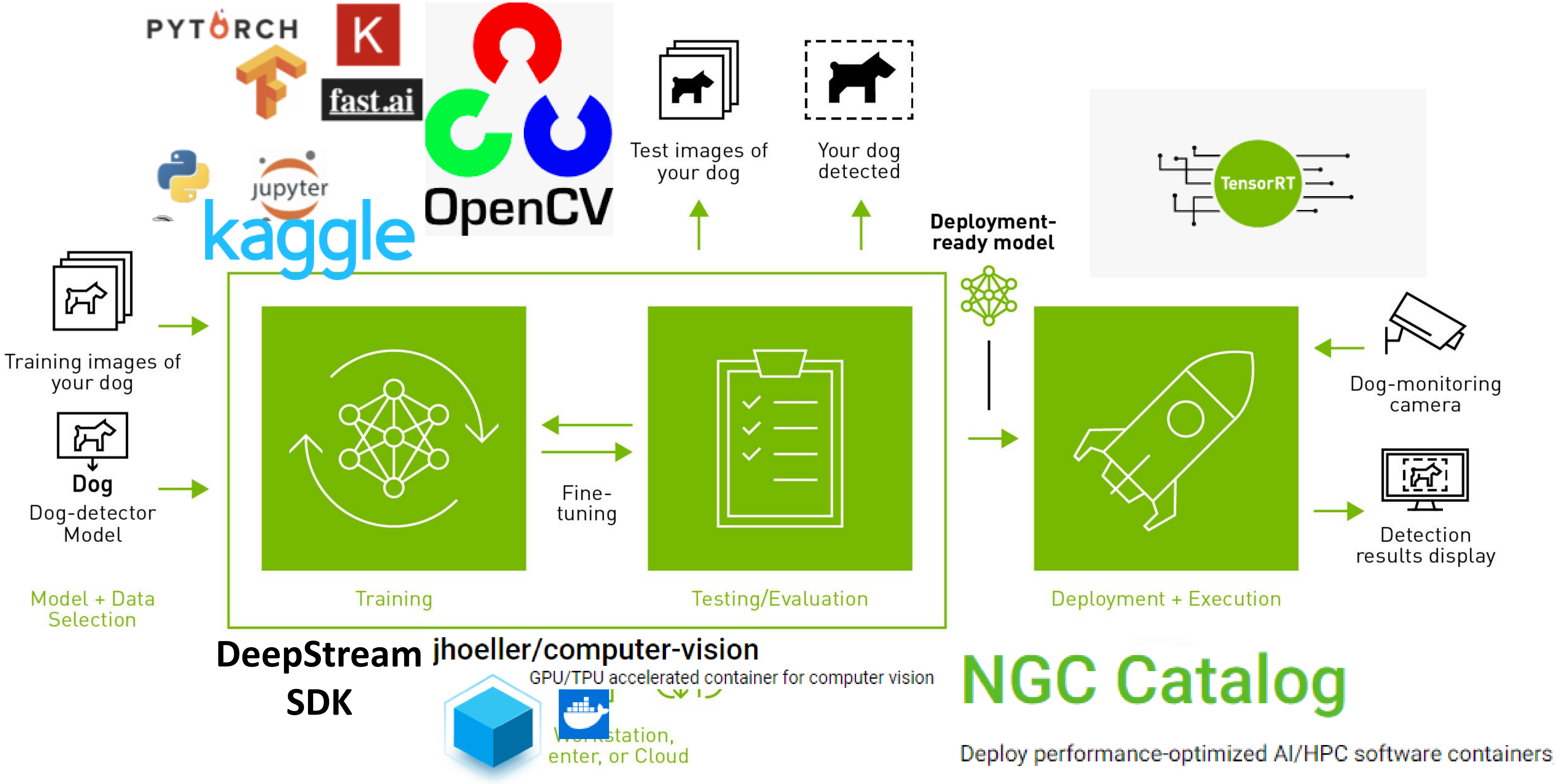

90
80
70
60
50
40
30
20
10
0
Zettabytes per year
Usable data created per year
Data Center traffic per year
22
31
40
52
67
85
7
9
12
14
17
21
Opportunity for Edge or Fog Computing
2016
2017
2018
2019
2020
2021


PlayStation
YouTube
Azure
aws
Server clusters
Alibaba Cloud
NETFLIX
XBOX
Google Cloud
STEAM
Tencent
facebook
Cloud
Unleash services to network edge
Base station
Edge node
Intelligent services
From edge to end
Edge
Edge computing network
End
End devices
Intelligent applications
Intelligent Edge
Edge Intelligence

Multi-access Edge Computing (MEC),

- Multi-access Edge Computing (MEC), also known as Mobile Edge Computing, is a key technology that enables mobile network operators to leverage edge-cloud benefits using their 5G networks.

- Following the concept of edge computing, MEC is located in close proximity to the connected devices and end-users and enables extremely low latency and high bandwidth while always enabling applications to leverage cloud capabilities as necessary.



Training images of your dog

Dog

Dog-detector Model

Model + Data Selection

Training

Fine-tuning

Testing/Evaluation

Test images of your dog

Your dog detected

Deployment-ready model

TensorRT

Deployment + Execution

Dog-monitoring camera

Detection results display

DeepStream SDK

jhoeller/computer-vision

GPU/TPU accelerated container for computer vision

Workstation,
enter, or Cloud

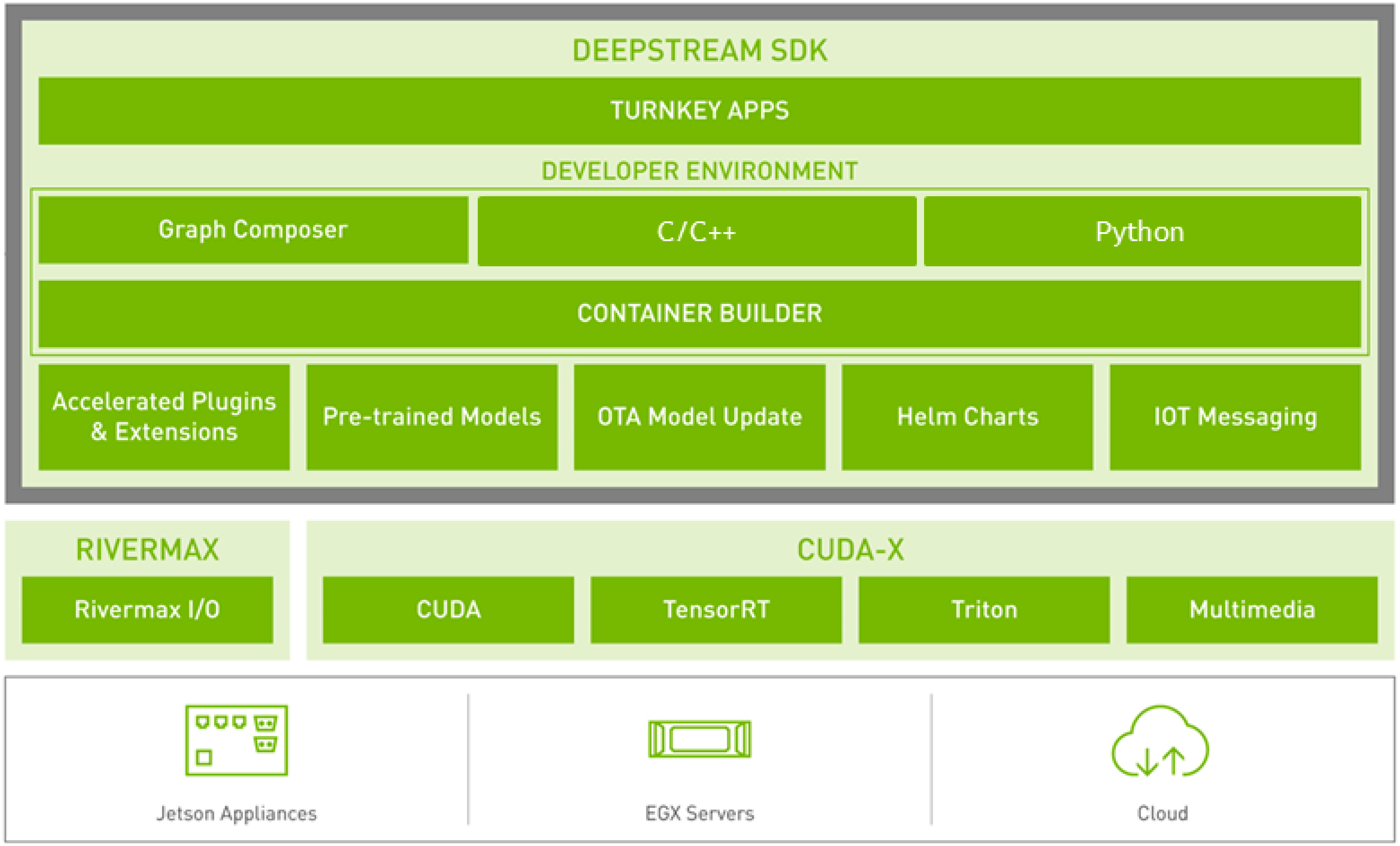
NGC Catalog

Deploy performance-optimized AI/HPC software containers

Deep Stream 6.2

- DeepStream is a streaming analytic toolkit to build AI-powered applications.
- It takes the streaming data as input - from USB/CSI camera, video from file or streams over RTSP?
- uses AI and computer vision to generate insights from pixels for better understanding of the environment.
- DeepStream SDK can be the foundation layer for a number of video analytic solutions like understanding traffic and pedestrians in smart city,

DEEPSTREAM SDK



DEEPSTREAM SDK

TURNKEY APPS

DEVELOPER ENVIRONMENT

Graph Composer

C/C++

Python

CONTAINER BUILDER

Accelerated Plugins & Extensions

Pre-trained Models

OTA Model Update

Helm Charts

IOT Messaging

RIVERMAX

Rivermax I/O

CUDA-X

CUDA

TensorRT

Triton

Multimedia

Jetson Appliances

EGX Servers

Cloud

Deepstream Demo

```
ssh –Y aijuser@dhruv
```

```
source go2deepstream
```

your current directory will be as follows:

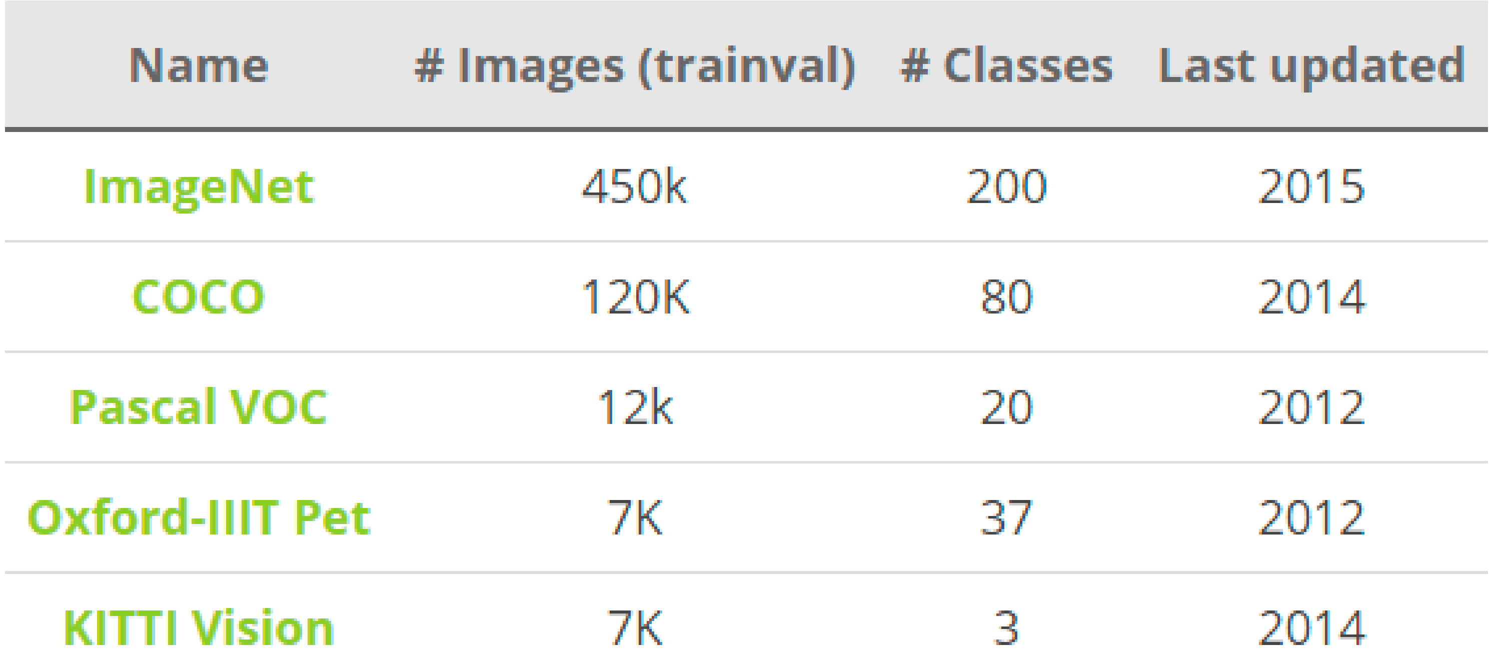
aijuser@dhruv:~/EDGE-AI/GSTREAM/opt/nvidia/deepstream/deepstream-6.2/samples/configs/deepstream-app

Execute the following:

```
$ deepstream-app -c source4_1080p_dec_infer-resnet_tracker_sgie_tiled_display_int8.yml
```

- Left click on the video to see the details
- Type q on the video to quit

Computer Vision Datasets



Name	# Images (trainval)	# Classes	Last updated
ImageNet	450k	200	2015
COCO	120K	80	2014
Pascal VOC	12k	20	2012
Oxford-IIIT Pet	7K	37	2012
KITTI Vision	7K	3	2014

Data set COCO (Common Objects in Context)

COCO is a large-scale object detection, segmentation, and captioning dataset.

- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints

https://cocodataset.org/#home

Pascal VOC (Visual Object Classes):

Large number of images collected from various sources, covering 20 object categories commonly encountered in everyday scenes

primarily used for object detection and segmentation tasks

- Pascal VOC is a dataset primarily used for object detection and segmentation tasks.
- The dataset provides detailed annotations, including object bounding boxes and class labels, making it suitable for training and evaluating object detection models.

Imagenet, a well known CV dataset

14 million images categorized into more than 20,000 object categories

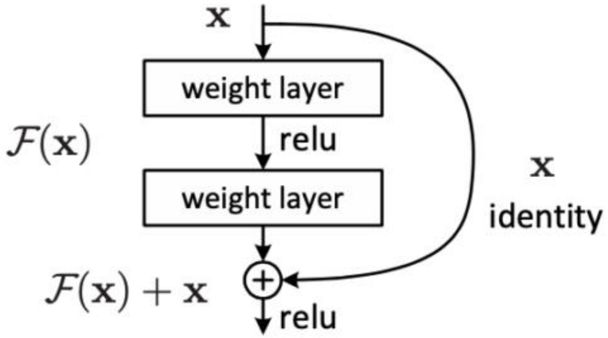
one of the largest and most well-known computer vision datasets.

- The ImageNet dataset covers a wide range of objects and scenes, and it has been widely used for various computer vision tasks, including object recognition and image classification.

Computer Vision Models of importance

ResNet, a Deep CNN

ResNet models have achieved state-of-the-art performance on various computer vision tasks, including image classification, object detection, and semantic segmentation



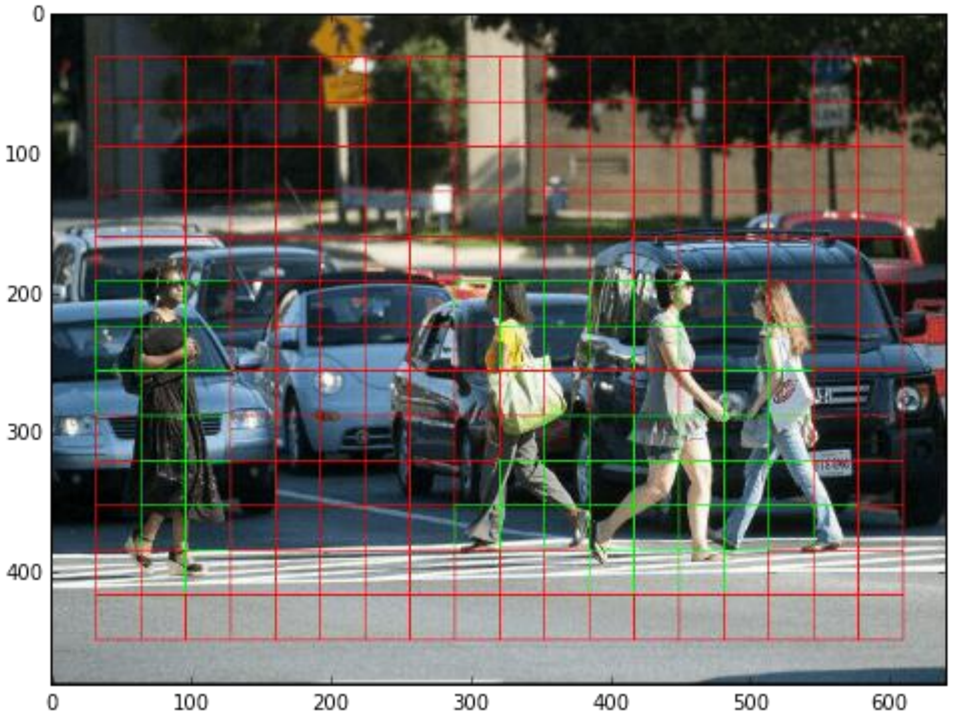
x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 2. Residual learning: a building block.

- ResNet is a deep convolutional neural network architecture used in computer vision.
- It introduced the concept of residual learning, which enables the training of very deep neural networks by addressing the vanishing gradient problem..

YOLO

Popular for applications requiring fast and accurate real-time object detection, such as autonomous vehicles, video surveillance, and robotics.



- real-time object detection model known for its speed and accuracy.
- Traditional object detection models use region proposal algorithms.
- YOLO takes a single network pass over the image and directly predicts bounding boxes and class probabilities for detected objects.

Mask R-CNN

Popular for pixel-level instance segmentation capabilities. It can detect objects and generate accurate object masks simultaneously

- Mask R-CNN (Region-based Convolutional Neural Network):
- Mask R-CNN is an extension of the Faster R-CNN object detection model, which also includes pixel-level instance segmentation capabilities. It can detect objects and generate accurate object masks simultaneously.
- Mask R-CNN has been widely adopted for various tasks, such as instance segmentation, object detection, and key point detection, where precise object boundaries and pixel-level details are required.

Use Cases of Computer Vision in Manufacturing

- Quality Control:
    - Defect detection is an essential part of the manufacturing process. When running an assembly line where millions of products are made, defects need to be caught in real time. Devices that use edge computing can make decisions in microseconds, catch defects instantly, and alert staff. This capability provides a significant advantage to factories as it can reduce waste and improve manufacturing efficiency.

- Worker Safety:
    - Industrial workers often operate heavy machinery and handle hazardous materials at manufacturing sites. Using a network of cameras and sensors equipped with AI-enabled video analytics, manufacturers can identify workers in unsafe conditions and quickly intervene to prevent accidents. Edge computing is critical to worker safety since life-saving decisions need to be made in real time.

Embedded Vision Camera

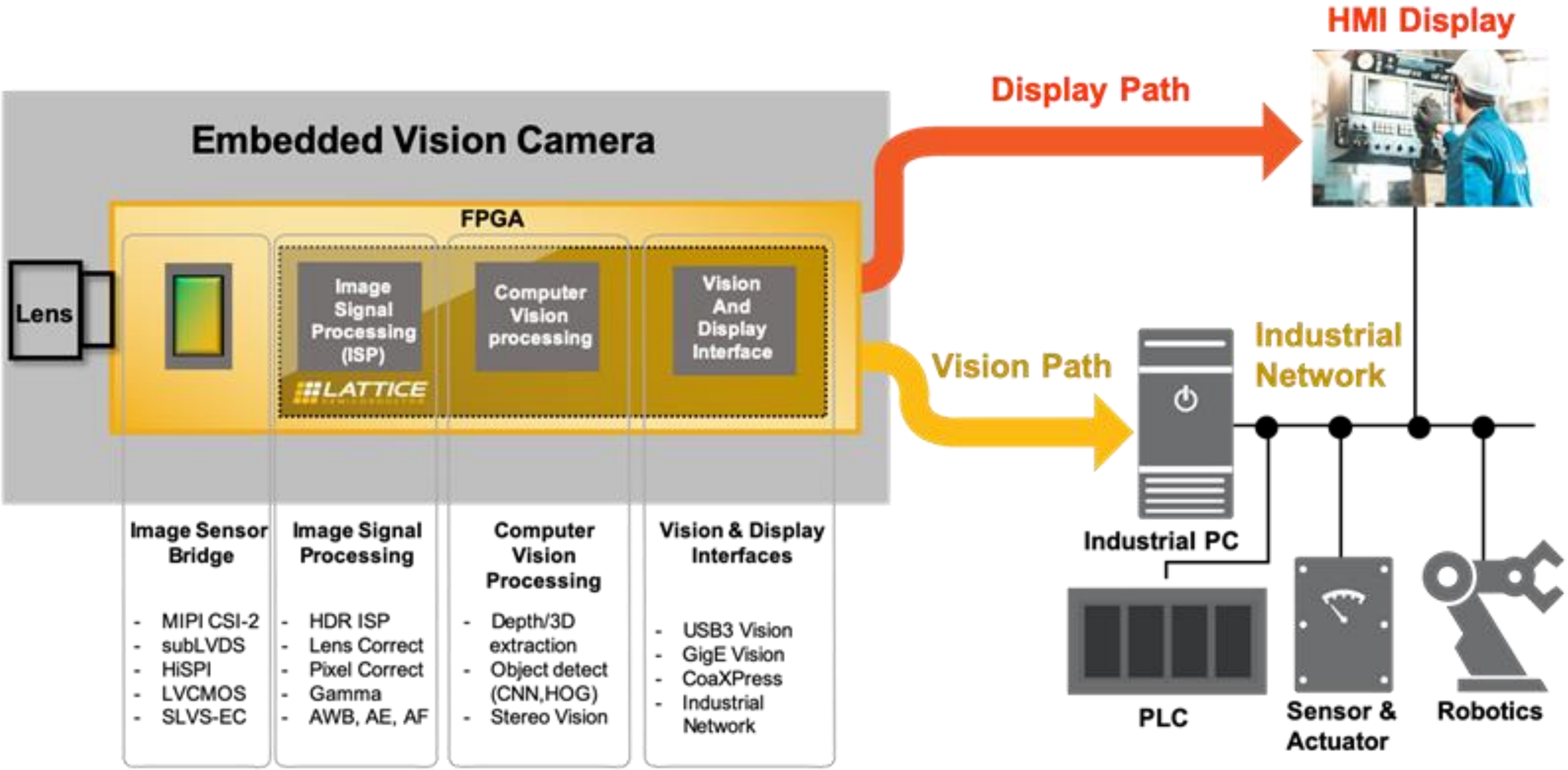


Image Sensor Bridge

- MIPI CSI-2
- subLVDS
- HiSPI
- LVCMOS
- SLVS-EC

Image Signal Processing

- HDR ISP
- Lens Correct
- Pixel Correct
- Gamma
- AWB, AE, AF

Computer Vision Processing

- Depth/3D extraction
- Object detect (CNN,HOG)
- Stereo Vision

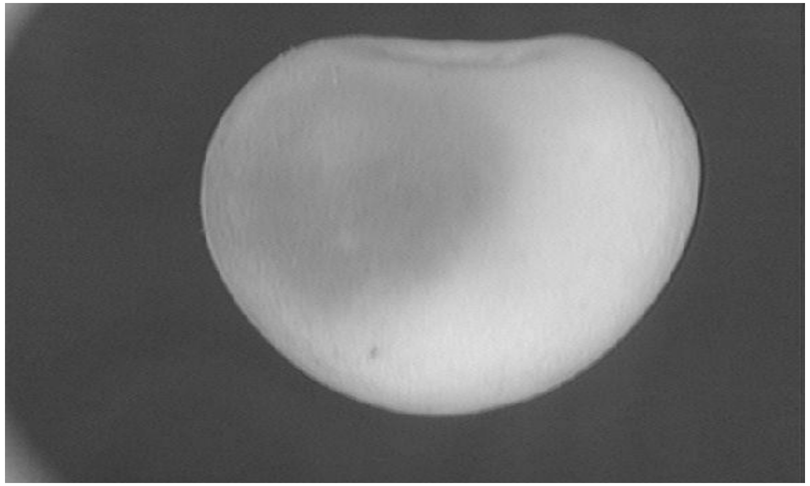
Vision & Display Interfaces

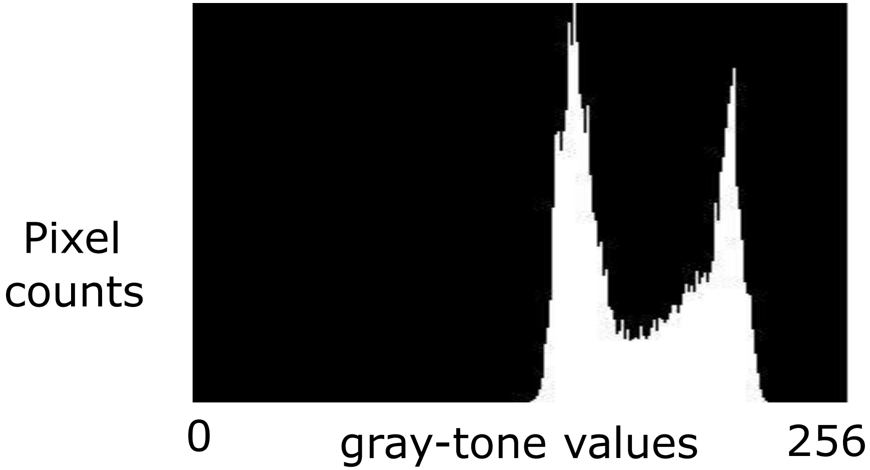
- USB3 Vision
- GigE Vision
- CoaXPress
- Industrial Network

Show and Tell Computer Vision

Thresholding example

- Observe the Cherry Fruit image
- Background is black
- Healthy fruit is bright
- Spoiled portion is medium dark
- Histogram shows two cherry regions (black background has been removed)





Thresholding Demo

login to aijuser@dhruv

```
ssh –Y aijuser@dhruv

source go2threshholding

cd /home/aijuser/EDGE-AI/THRESH-HOLDING

source go-run
```

You will see various image windows processed using different thresholding algorithm

How-to-check-installed-Python-version?

Check your Python version

```
python --version
python 3.8.3
```

ensure it is at least Python 3.7 or above

How-to-install-opencv-for-python-in-linux

```
pip install opencv-contrib-python
```

```
Defaulting to user installation because normal site-packages is not writeable
Collecting opencv-contrib-python
Using cached opencv_contrib_python-4.5.2.52-cp36-cp36m-
manylinux2014_x86_64.whl (57.4 MB)
Requirement already satisfied: numpy>=1.13.3 in
/home/sambath/.local/lib/python3.6/site-packages (from opencv-contrib-python)
(1.19.5)
Installing collected packages: opencv-contrib-python
Successfully installed opencv-contrib-python-4.5.2.52
```

This installs main module as well as community-contributed module

For more details:

https://www.geeksforgeeks.org/how-to-install-opencv-for-python-in-linux/

How-to-check-installed-opencv-version?

```
python
>>>import cv2
>>>cv2.__version__
‘4.5.1’
ctrl+D
```

Play a video using the OpenCV Python

```
# importing libraries
import cv2
import numpy as np
# Create a VideoCapture object and read from input file
cap = cv2.VideoCapture(‘dog.mp4')
# Check if camera opened successfully
if (cap.isOpened()== False):
print("Error opening video file")
# Read until video is completed
while(cap.isOpened()):
# Capture frame-by-frame
ret, frame = cap.read()
if ret == True:
        # Display the resulting frame
        cv2.imshow('Frame', frame)
        # Press Q on keyboard to exit
        if cv2.waitKey(25) & 0xFF == ord('q'):
        break
# Break the loop
else:
        break
# When everything done, release
# the video capture object
cap.release()
# Closes all the frames
cv2.destroyAllWindows()
```

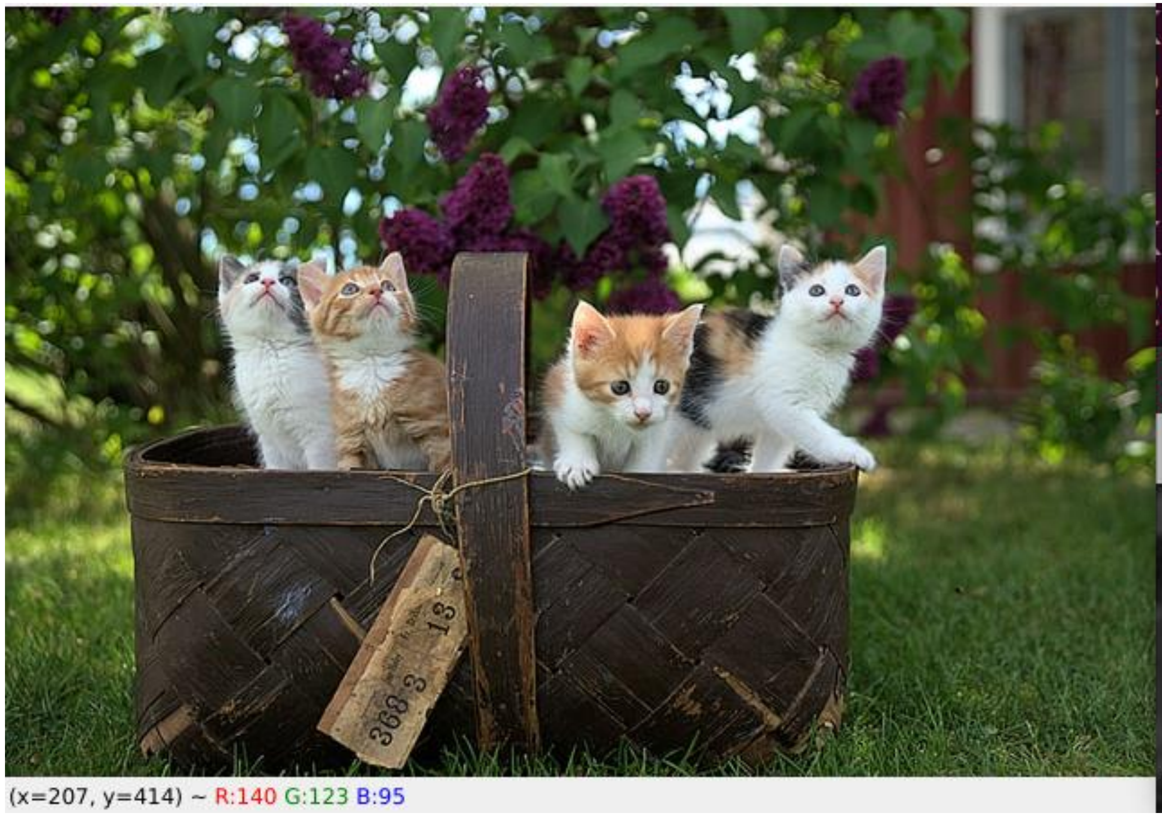
Image Contouring demo

```
login to aijuser@dhruv

source go2contour

python3 contours.py
```

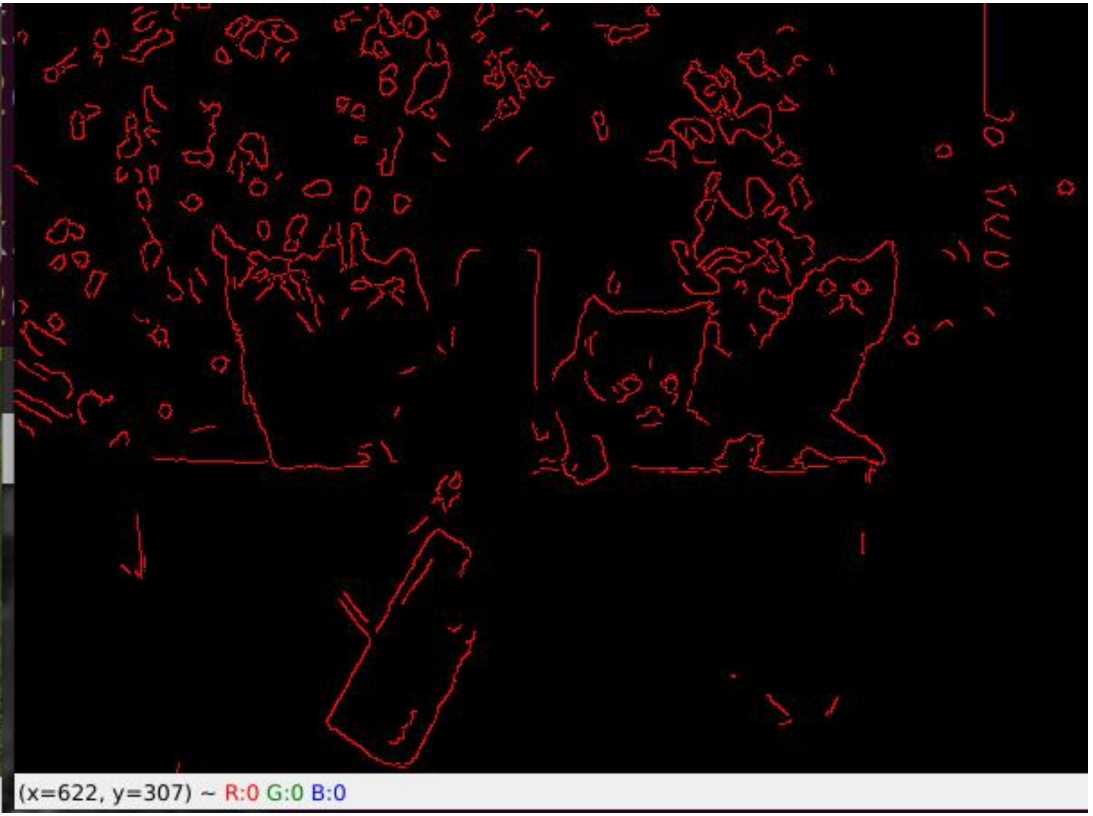
What is an image contour?

Identify structural outlines of objects in an image

Helps in identifying the shape of the object



```
(x=207, y=414) ~ R:140 G:123 B:95
```



```
(x=622, y=307) ~ R:0 G:0 B:0
```

Bit wise operations in CV

- Bitwise operations can be used in image manipulations
- These bitwise techniques are used in many computer vision applications like for
  - *creating masks of the image*
  - *adding watermarks to the image*
  - *creating a new image using these bitwise operators*

Bit wise operations in CV

- These operations work on the individual pixels in the image to give accurate results compared with other morphing techniques in OpenCV
  - Bitwise AND
    - pixels that are common with image 1 and image 2 and remaining pixels are removed from the output
  - Bitwise OR
    - disjunction of the pixels in both images.we will not eliminate pixels but it merges both images.
  - Bitwise NOT
    - replaces the white pixels with black pixels and vice versa
  - Bitwise XOR
    - inverts the pixels which are intersected in image 1 & image 2 and the rest of the pixels remains the same.

Bit wise operations

cd ~/EDGE-AI/COMPUTER-VISION/opencv-course-master/Section2-Advanced

python3 masking_new.py

code snippet - portion of python script

```
…
# cv2.bitwise_xor is applied over the
# image inputs with applied parameters
# uncomment corresponding dest_* & cv2.imshow statements
#dest_xor = cv2.bitwise_xor(img1, img2, mask = None)
#dest_and = cv2.bitwise_and(img1, img2, mask = None)
#dest_not = cv2.bitwise_not(img1, img2, mask = None)
dest_or = cv2.bitwise_or(img1, img2, mask = None)
# the window showing output image
# with the Bitwise XOR operation
# on the input images
#cv2.imshow('Bitwise XOR', dest_xor)
#cv2.imshow('Bitwise AND', dest_and)
#cv2.imshow('Bitwise NOT', dest_not)
cv2.imshow('Bitwise OR', dest_or)
```

Image transformations

```
cd ~/EDGE-AI/COMPUTER-VISION/opencv-course-master/Section1-Basics

python3 transformations.py
```

Transformation using Opencv

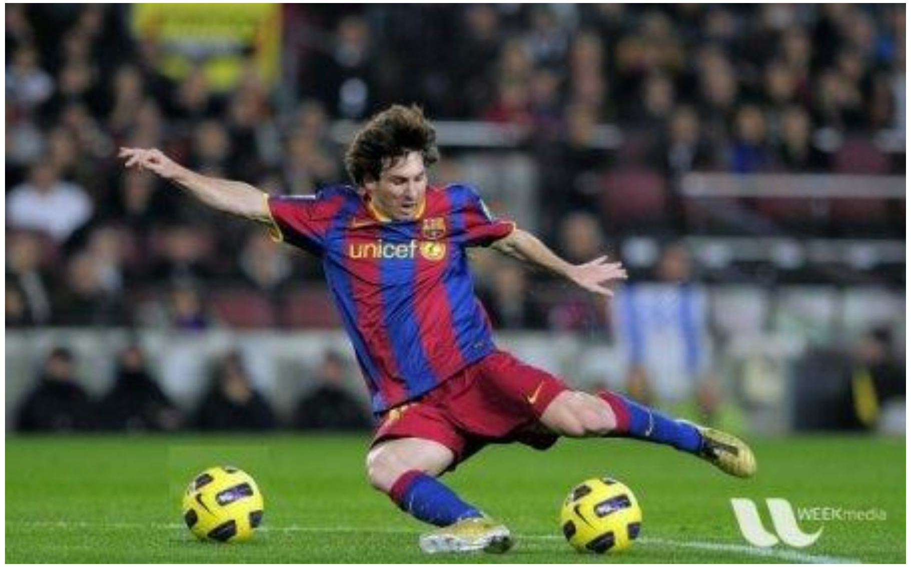
```
#pylint:disable=no-member
           import cv2 as cv
      import numpy as np
      img = cv.imread('/home/sambath/EDGE-AI/ArmeniaCat.jpg')
       cv.imshow('Park', img)
       # Translation
     def translate(img, x, y):
          transMat = np.float32([[1,0,x],[0,1,y]])
          dimensions = (img.shape[1], img.shape[0])
          return cv.warpAffine(img, transMat, dimensions)
           # -x --> Left
     # -y --> Up
      # x --> Right
      # y --> Down
     translated = translate(img, -100, 100)
     cv.imshow('Translated', translated)
         # Rotation
     def rotate(img, angle, rotPoint=None):
         (height,width) = img.shape[:2]
```

```
if rotPoint is None:
             rotPoint = (width//2,height//2)
          rotMat = cv.getRotationMatrix2D(rotPoint, angle, 1.0)
          dimensions = (width,height)
          return cv.warpAffine(img, rotMat, dimensions)
       rotated = rotate(img, -45)
      cv.imshow('Rotated', rotated)
      rotated_rotated = rotate(img, -90)
      cv.imshow('Rotated Rotated', rotated_rotated)
      # Resizing
      resized = cv.resize(img, (500,500),
interpolation=cv.INTER_CUBIC)
      cv.imshow('Resized', resized)
     # Flipping
      flip = cv.flip(img, -1)
      cv.imshow('Flip', flip)
      # Cropping
      cropped = img[200:400, 300:400]
        cv.imshow('Cropped', cropped)
        cv.waitKey(0)
```

Image ROI – Region of Interest

- Sometimes, you will have to play with certain regions of images. For eye detection in images, first face detection is done over the entire image. When a face is obtained, we select the face region alone and search for eyes inside it instead of searching the whole image. It improves accuracy (because eyes are always on faces :D) and performance (because we search in a small area).
- ROI is again obtained using Numpy indexing. Here I am selecting the ball and copying it to another region in the image:

```
>>> ball = img[280:340, 330:390]
>>> img[273:333, 100:160] = ball
```



Demo Image ROI - Region of Interest

```
cd ~/EDGE-AI/WORKING
python3 messi2_roi.py;clear
```

Select Region of Interest and ENTER

Separate wind showing ROI will be displayed

Click on ROI window and ENTER

Capture Video from your webcam

```
cd ~/EDGE-AI/WORKING

python3 vid-cap.py;clear

PRESS‘q’in video frameto exit
```

```
import numpy as np
import cv2 as cv
cap = cv.VideoCapture(0)
if not cap.isOpened():
    print("Cannot open camera")
    exit()
while True:
    # Capture frame-by-frame
    ret, frame = cap.read()
    # if frame is read correctly ret is True
    if not ret:
        print("Can't receive frame (stream end?)
Exiting ...")
        break
    # Our operations on the frame come here
    gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
    # Display the resulting frame
    cv.imshow('frame', gray)
    if cv.waitKey(1) == ord('q'):
        break
# When everything done, release the capture
cap.release()
cv.destroyAllWindows()
```

Face Detection

- Face detection is a type of application.
- It is the process in which algorithms are developed and trained to properly locate faces), in images.
- These can be in real time from a video camera or from photographs.
- In order to recognize a face, the camera software must first detect it and identify the features before making an identification.
- Examples where Face detection is used are in
  1. Airport security systems. Likewise
  2. Facebook makes tagging suggestions to identify people in photos it must first locate the face.
  3. On Snapchat, face detection is required to augment reality which allows users to virtually wear dog face masks using fancy filters
  4. Another use of face detection is in smartphone face ID security.

Source:
https://becominghuman.ai/face-detection-using-opencv-with-haar-cascade-classifiers-941dbb25177

Face Detection and Face Recognition Demo

Face Detection in Python Using your Webcam

(if you are using your camera for videocall, this program won’t work)

Login to aijuser@dhruv

```
cd
```

```
Source go2faceinwebcam
source go-run
```

Webcam will open and faces captured by camera will be detected

CTRL+c to exit

Source:
https://realpython.com/face-detection-in-python-using-a-webcam/

Face Detection in Opencv - Python

```
login to aijuser@dhruv

source go2facedetection

source go-run

Group of people faces are recognized

To Exit …
keep cursor on the photo type q
```

Face Recognition in Opencv - Python

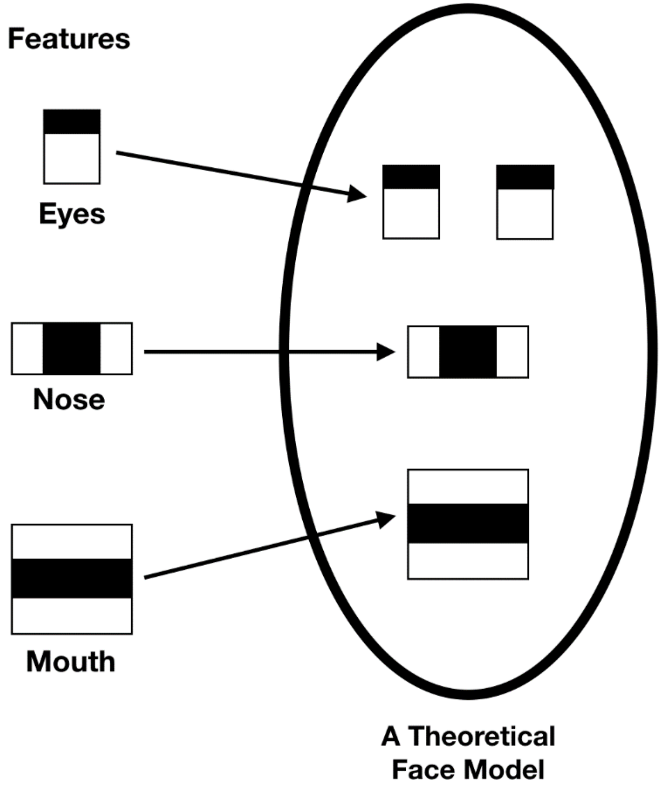
```
login to aijuser@dhruv

 source go2facerecognition

 source go-run-recognize-faces

 Training done ……………. . .

 Select a Face-ID to work on?  - Type Enter

 Type either 0 or 1 or 2

 A face appears and you hit Enter that face is recognized

 Again run source go-run-recognize-faces
```

Understanding Haar Cascades

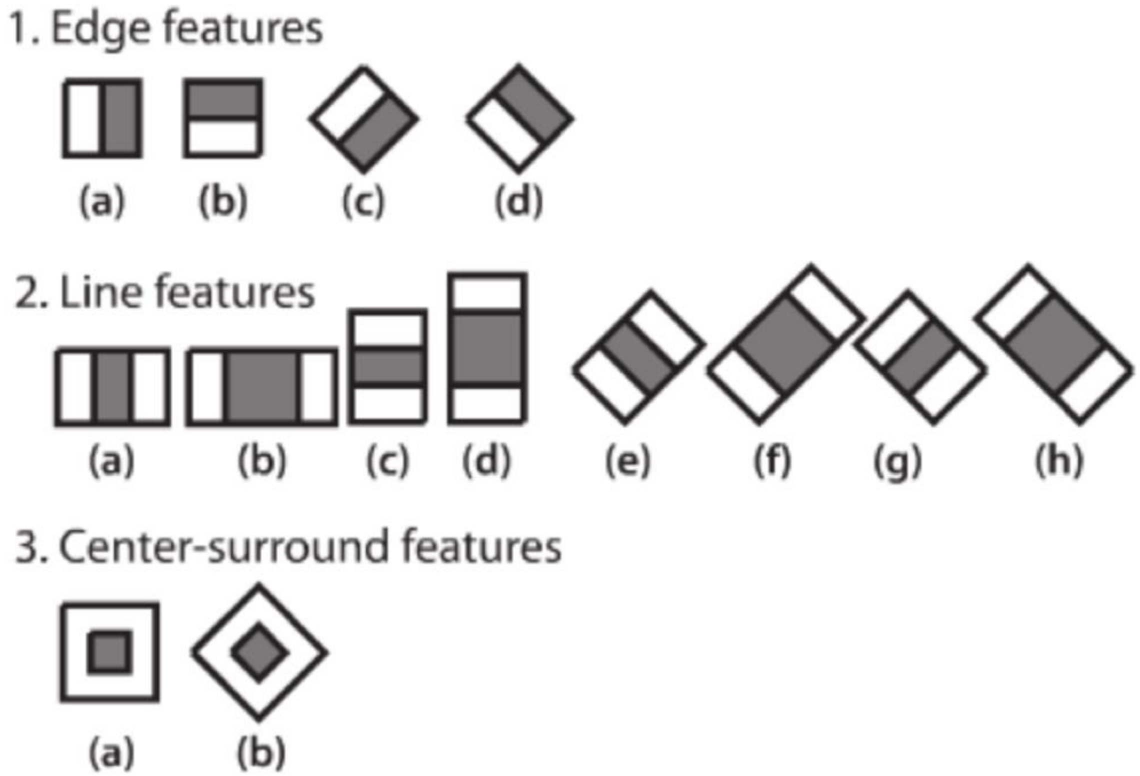
- *Haar Cascade* is based on “Haar Wavelets”
- A Haar Wavelets is a sequence of rescaled “square-shaped” functions which together form a wavelet family or basis.
- It is based on the Haar Wavelet technique to analyze pixels in the image into squares by function.
- This uses ML algorithm to get a high degree of accuracy from “training data”.
- This uses “integral image” concepts to compute the “features” detected.
- Haar Cascades use the **Adaboost** learning algorithm which selects a small number of important features from a large set to give an efficient result of classifiers.

Face Model

- A theoretical face Model



Difference between Face detection and Face Recognition



Face Detection determines the locations and sizes of human faces in arbitrary (digital) images.

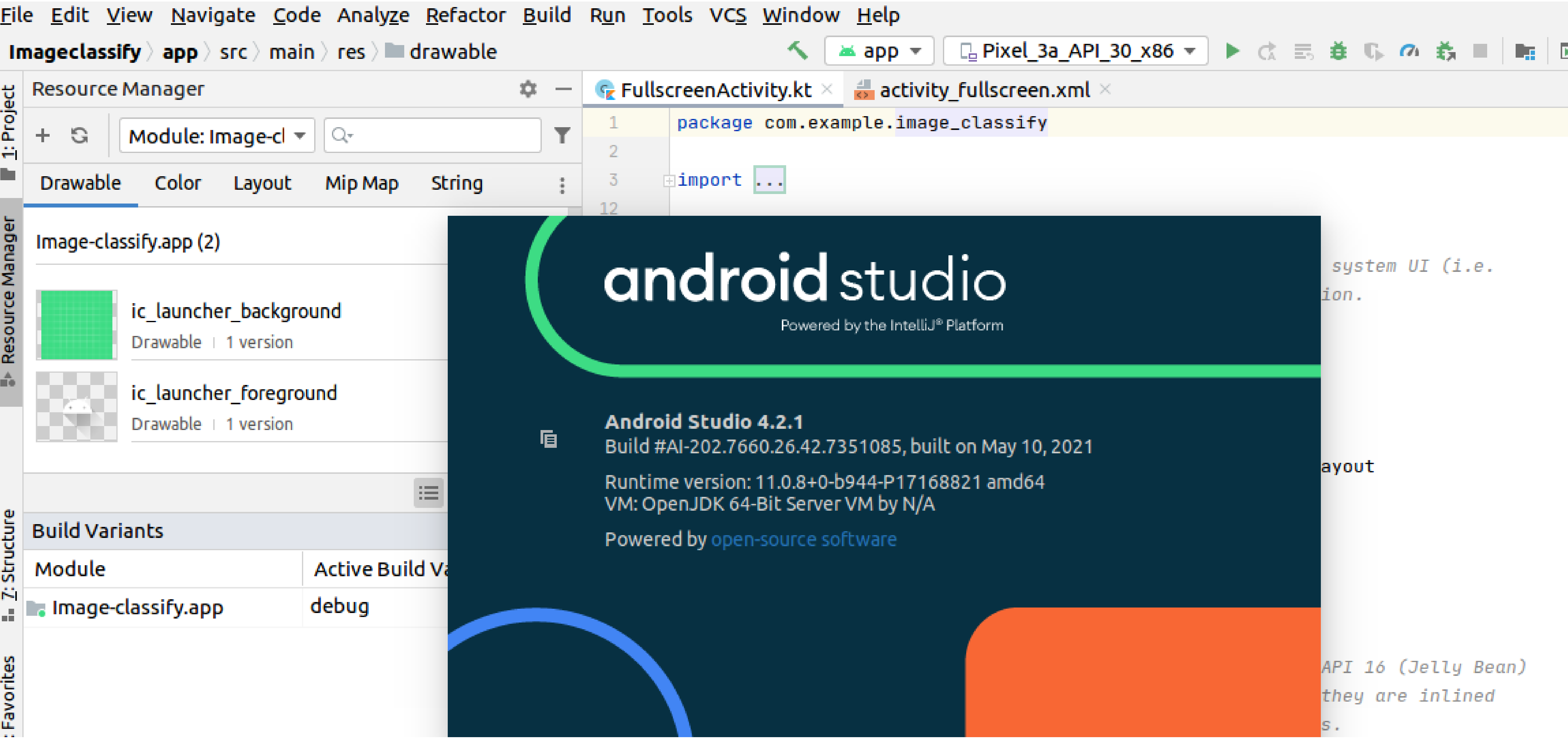
In **Face Recognition**, the use of Face Detection comes first to determine and isolate a face before it can be recognized.

Object Recognition Demo

sambath.narayanan@gmail.com

Object Recognition

- Object recognition (also called **object classification**) – one or several pre-specified or learned objects or object classes can be recognized, usually together with their 2D positions in the image or 3D poses in the scene..
- **Identification** – an individual instance of an object is recognized. Examples include identification of a specific person's face or fingerprint, identification of handwritten digits, or identification of a specific vehicle.
- Detection – the image data are scanned for a specific condition.
  - Examples
    - detection of possible abnormal cells or tissues in medical images
    - detection of a vehicle in an automatic road toll system


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Imageclassify 〉 app 〉 src 〉 main 〉 res 〉 drawable
app
Pixel_3a_API_30_x86
Resource Manager
Module: Image-cl
Drawable Color Layout Mip Map String
FullscreenActivity.kt
activity_fullscreen.xml
1: Project
Resource Manager
7: Structure
: Favorites
android studio
Powered by the IntelliJ® Platform

```
package com.example.image_classify

import ...
```

```
system UI (i.e.
ion.
ayout
API 16 (Jelly Bean)
they are inlined
s.
```

Image-classify.app (2)

- ic_launcher_background — Drawable | 1 version
- ic_launcher_foreground — Drawable | 1 version

Build Variants

Module	Active Build Va
Image-classify.app	debug

Android Studio 4.2.1
Build #AI-202.7660.26.42.7351085, built on May 10, 2021

Runtime version: 11.0.8+0-b944-P17168821 amd64
VM: OpenJDK 64-Bit Server VM by N/A

Powered by open-source software

Object Detection

(I). Login to gmeet from Samson M2 – 8 Cores

(II). Share the entire screen

(III)Open Tensor Flow Lite App and recognize objects

Hand Pose Detection

sambath.narayanan@gmail.com

Hand Pose Detection

(I). Login to 192.168.0.113(cvraman)

(II). Type the command in the Terminal window ‘lsusb’

(III). Ensure there is a Camera device listed as follows. Other USB devices will also be shown.

```
Bus 001 Device 002: ID 046d:0825 Logitech, Inc. Webcam C270
```

(IV). Start the docker Container

In the jetcam folder there is Sambath-Camera.ipynb, run that in Jupyter

First cell command will show /dev/video0
(while running subsequent cells verify the previous cmd is complete)

MUST RUN Last cell – after successfully running all cells
Os.__exit(00)

From trt_pose_hand, take live_hand_pose.ipynb and execute every cell

Hand Pose Detection

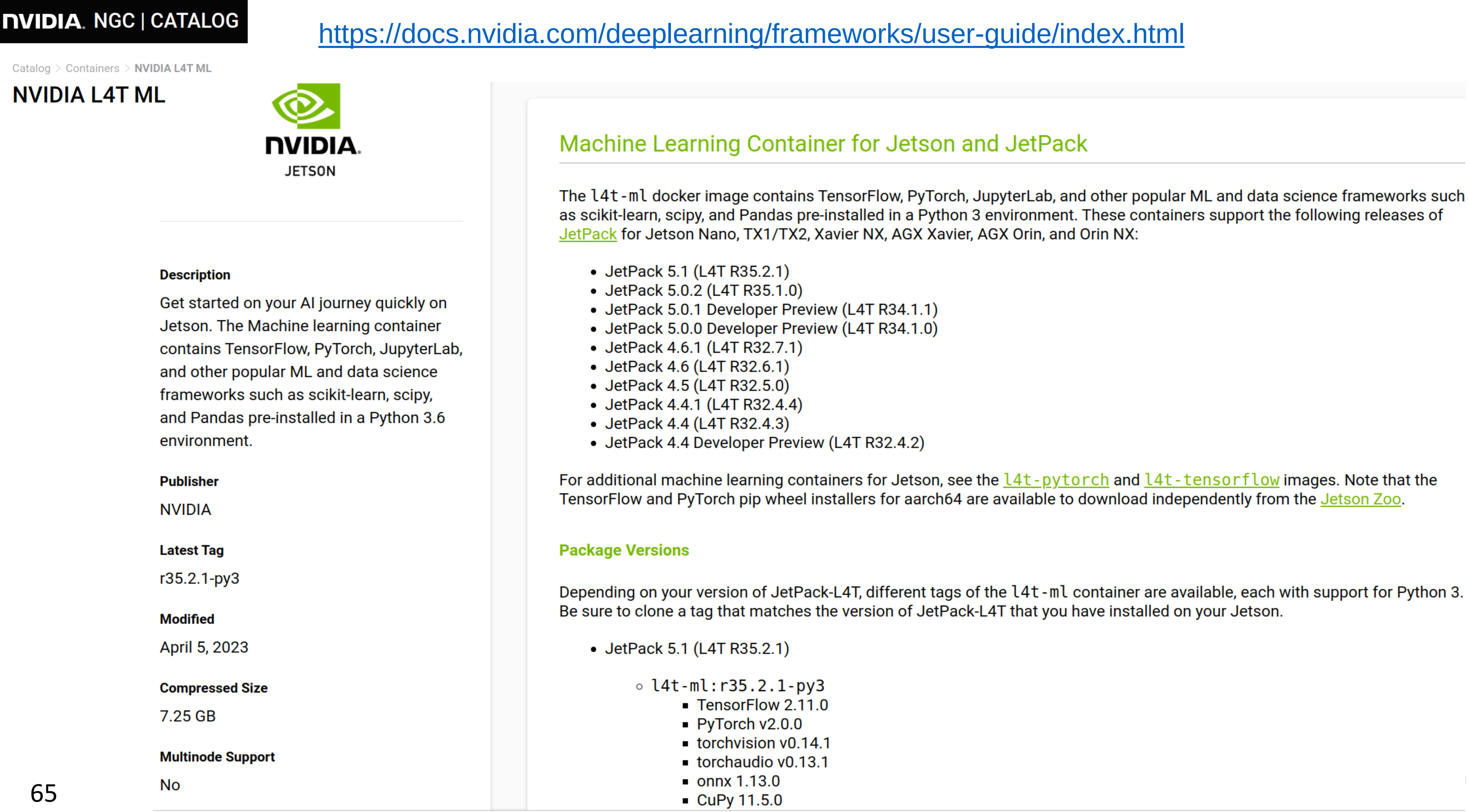
(I). Download the following PreTrained Model Weights
Hand_pose_resnet18_baseline_224x224

(2).
The current gesture classification model supports
Pan,stop,fine,fist,peace, and no hand

(3). To make your own gesture classification

Create your own data set using gesture data collection
.ipynb

https://docs.nvidia.com/deeplearning/frameworks/user-guide/index.html



Catalog > Containers > NVIDIA L4T ML

NVIDIA L4T ML

Description

Get started on your AI journey quickly on Jetson. The Machine learning container contains TensorFlow, PyTorch, JupyterLab, and other popular ML and data science frameworks such as scikit-learn, scipy, and Pandas pre-installed in a Python 3.6 environment.

Publisher

NVIDIA

Latest Tag

r35.2.1-py3

Modified

April 5, 2023

Compressed Size

7.25 GB

Multinode Support

No

Machine Learning Container for Jetson and JetPack

The `l4t-ml` docker image contains TensorFlow, PyTorch, JupyterLab, and other popular ML and data science frameworks such as scikit-learn, scipy, and Pandas pre-installed in a Python 3 environment. These containers support the following releases of JetPack for Jetson Nano, TX1/TX2, Xavier NX, AGX Xavier, AGX Orin, and Orin NX:

- JetPack 5.1 (L4T R35.2.1)
- JetPack 5.0.2 (L4T R35.1.0)
- JetPack 5.0.1 Developer Preview (L4T R34.1.1)
- JetPack 5.0.0 Developer Preview (L4T R34.1.0)
- JetPack 4.6.1 (L4T R32.7.1)
- JetPack 4.6 (L4T R32.6.1)
- JetPack 4.5 (L4T R32.5.0)
- JetPack 4.4.1 (L4T R32.4.4)
- JetPack 4.4 (L4T R32.4.3)
- JetPack 4.4 Developer Preview (L4T R32.4.2)

For additional machine learning containers for Jetson, see the `l4t-pytorch` and `l4t-tensorflow` images. Note that the TensorFlow and PyTorch pip wheel installers for aarch64 are available to download independently from the Jetson Zoo.

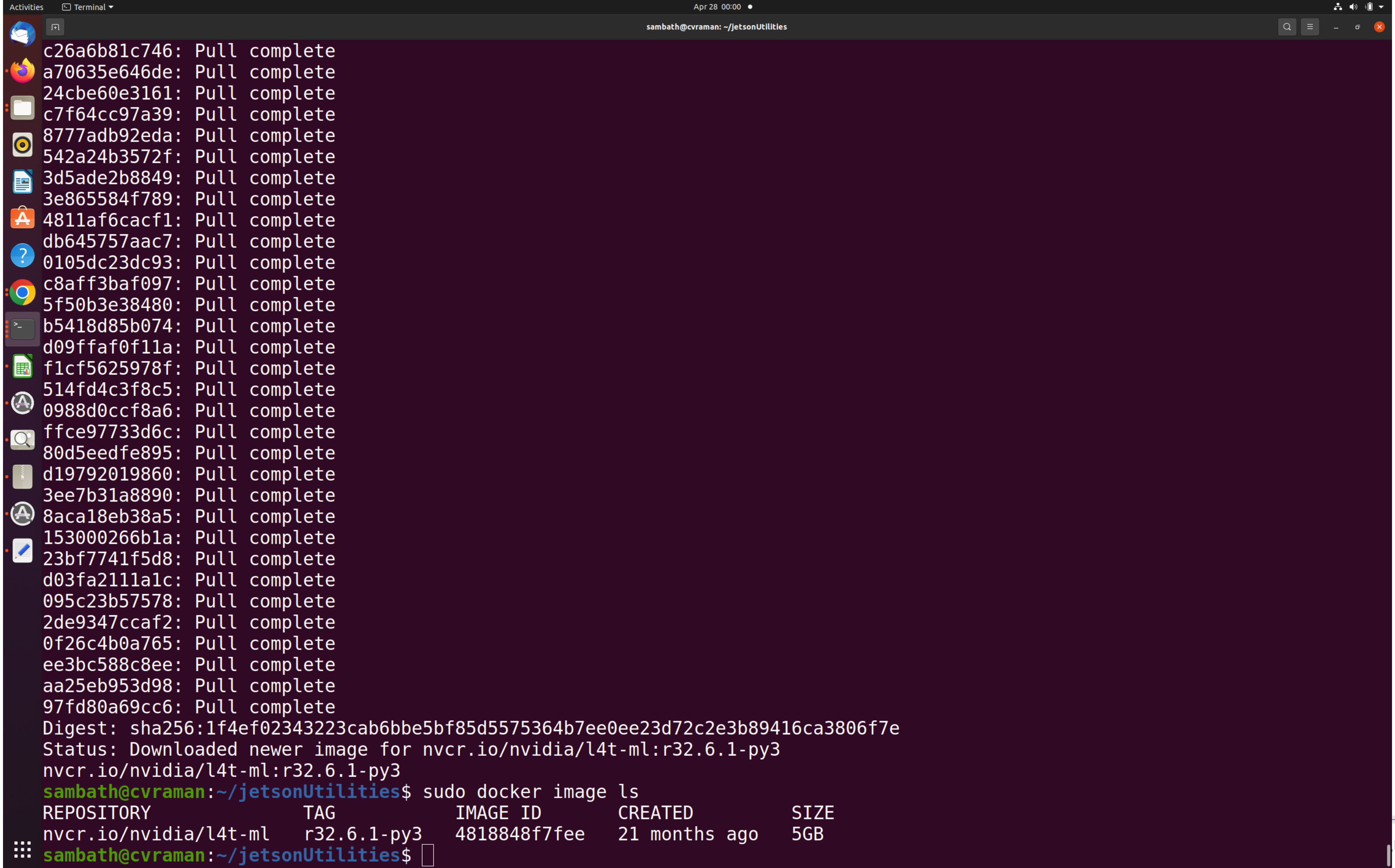
Package Versions

Depending on your version of JetPack-L4T, different tags of the `l4t-ml` container are available, each with support for Python 3. Be sure to clone a tag that matches the version of JetPack-L4T that you have installed on your Jetson.

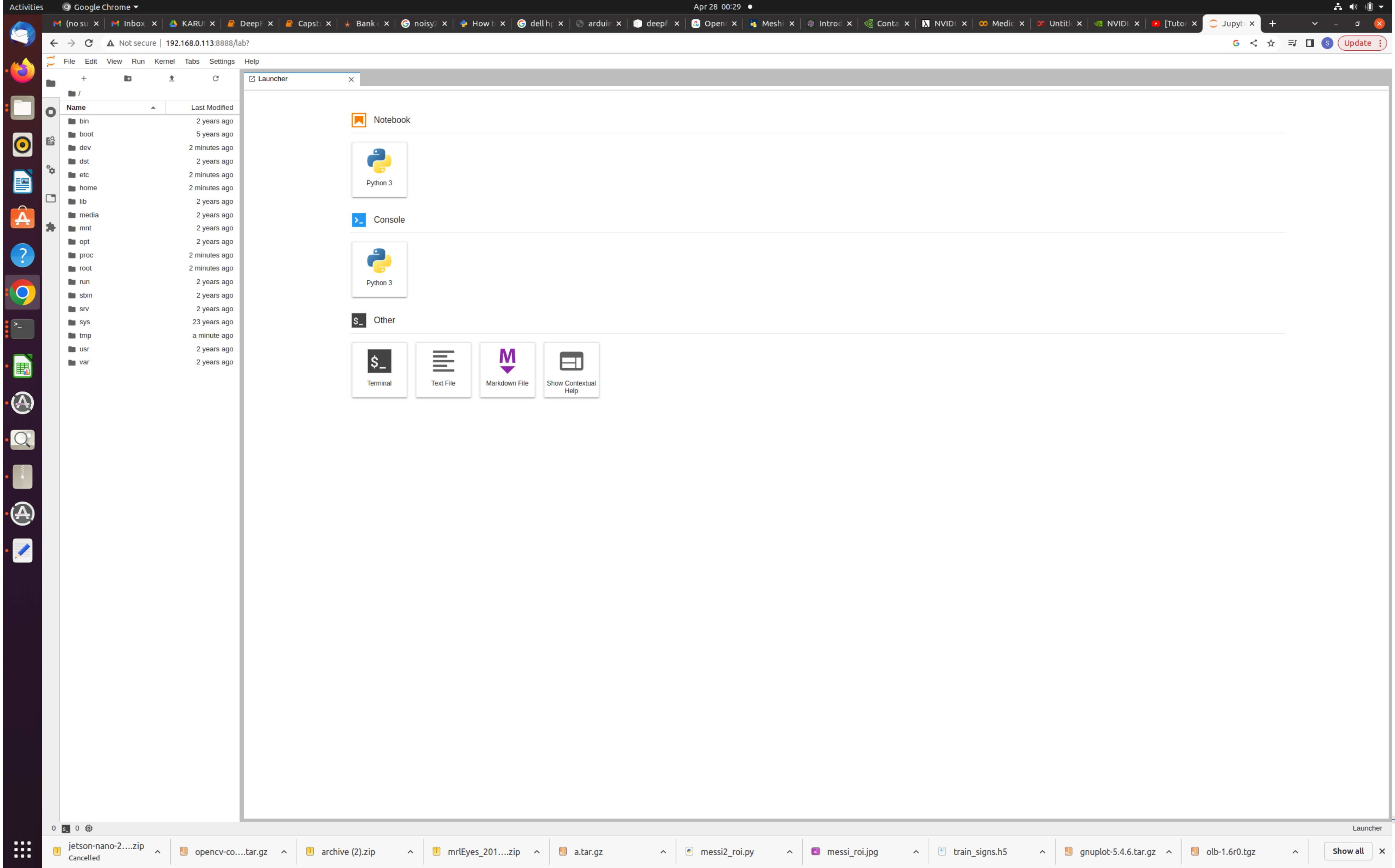
- JetPack 5.1 (L4T R35.2.1)
  - `l4t-ml:r35.2.1-py3`
    - TensorFlow 2.11.0
    - PyTorch v2.0.0
    - torchvision v0.14.1
    - torchaudio v0.13.1
    - onnx 1.13.0
    - CuPy 11.5.0

Summary of Hardware and software setup

- Prepare Jetson Nano with latest Jetpack SD Card image
  - Standalone Headless mode
- Pull the l4t-ml container
- Run the container and login to the Jupyter Notebook running on Jetson Nano
  - browser runs on Linux Desktop (192.168.0.113:8888)



```
c26a6b81c746: Pull complete
a70635e646de: Pull complete
24cbe60e3161: Pull complete
c7f64cc97a39: Pull complete
8777adb92eda: Pull complete
542a24b3572f: Pull complete
3d5ade2b8849: Pull complete
3e865584f789: Pull complete
4811af6cacf1: Pull complete
db645757aac7: Pull complete
0105dc23dc93: Pull complete
c8aff3baf097: Pull complete
5f50b3e38480: Pull complete
b5418d85b074: Pull complete
d09ffaf0f11a: Pull complete
f1cf5625978f: Pull complete
514fd4c3f8c5: Pull complete
0988d0ccf8a6: Pull complete
ffce97733d6c: Pull complete
80d5eedfe895: Pull complete
d19792019860: Pull complete
3ee7b31a8890: Pull complete
8aca18eb38a5: Pull complete
153000266b1a: Pull complete
23bf7741f5d8: Pull complete
d03fa2111a1c: Pull complete
095c23b57578: Pull complete
2de9347ccaf2: Pull complete
0f26c4b0a765: Pull complete
ee3bc588c8ee: Pull complete
aa25eb953d98: Pull complete
97fd80a69cc6: Pull complete
Digest: sha256:1f4ef02343223cab6bbe5bf85d5575364b7ee0ee23d72c2e3b89416ca3806f7e
Status: Downloaded newer image for nvcr.io/nvidia/l4t-ml:r32.6.1-py3
nvcr.io/nvidia/l4t-ml:r32.6.1-py3
sambath@cvraman:~/jetsonUtilities$ sudo docker image ls
REPOSITORY            TAG            IMAGE ID       CREATED         SIZE
nvcr.io/nvidia/l4t-ml   r32.6.1-py3   4818848f7fee   21 months ago   5GB
sambath@cvraman:~/jetsonUtilities$ 
```

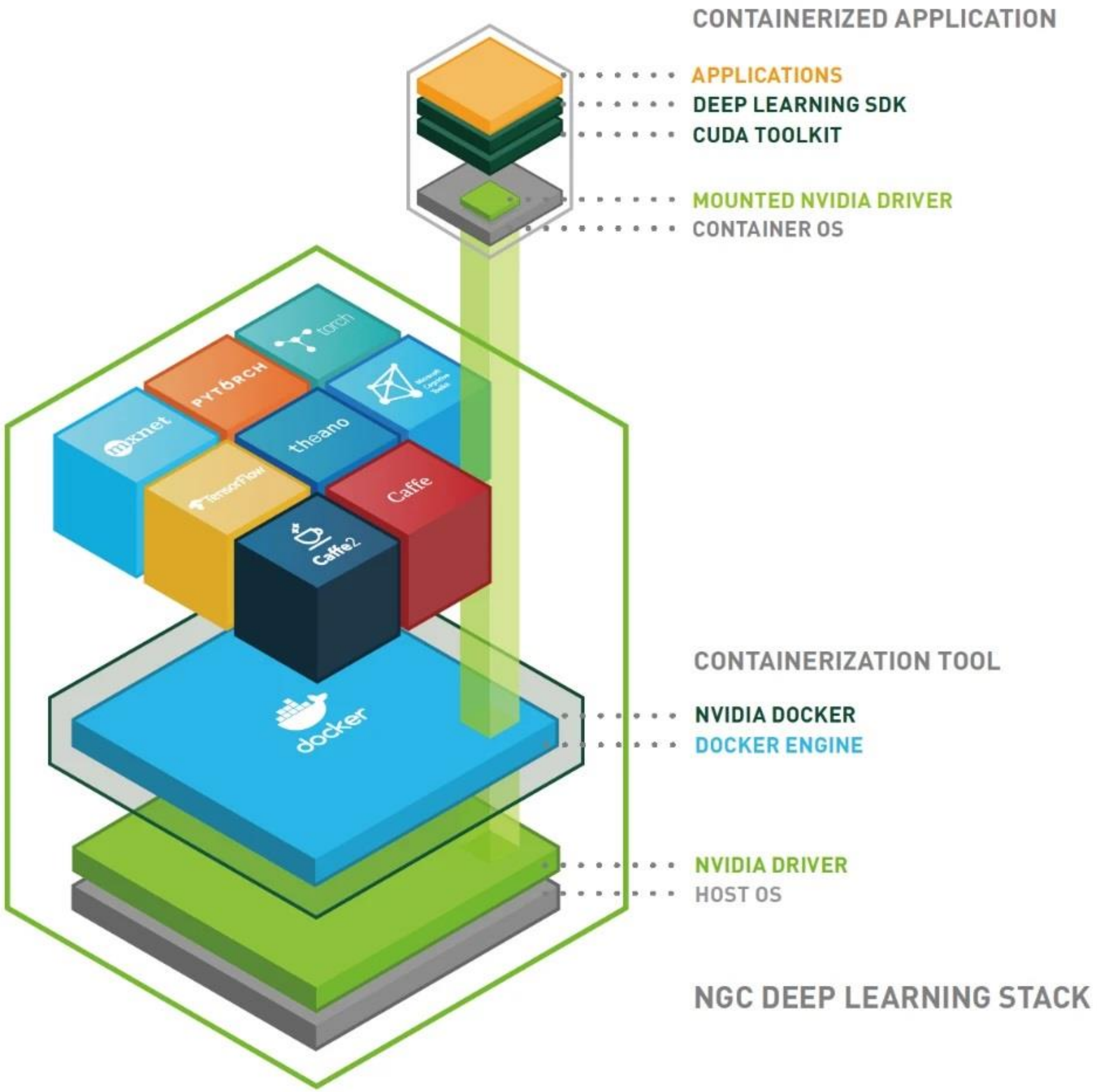


/

Name	Last Modified
bin	2 years ago
boot	5 years ago
dev	2 minutes ago
dst	2 years ago
etc	2 minutes ago
home	2 minutes ago
lib	2 years ago
media	2 years ago
mnt	2 years ago
opt	2 years ago
proc	2 minutes ago
root	2 minutes ago
run	2 years ago
sbin	2 years ago
srv	2 years ago
sys	23 years ago
tmp	a minute ago
usr	2 years ago
var	2 years ago

jetson-nano-2....zip Cancelled · opencv-co....tar.gz · archive (2).zip · mrlEyes_201....zip · a.tar.gz · messi2_roi.py · messi_roi.jpg · train_signs.h5 · gnuplot-5.4.6.tar.gz · olb-1.6r0.tgz

NGC Architecture



CONTAINERIZED APPLICATION

APPLICATIONS

DEEP LEARNING SDK

CUDA TOOLKIT

MOUNTED NVIDIA DRIVER

CONTAINER OS

CONTAINERIZATION TOOL

NVIDIA DOCKER

DOCKER ENGINE

NVIDIA DRIVER

HOST OS

NGC DEEP LEARNING STACK

Analyze Classify Chest XRAY

sambath.narayanan@gmail.com

COVID XRAY classification

Login to Google in dataeverconsulting user account

Then open colab

From Colab open the covid.ipynb file

Mount the Dataeverconsulting Google Drive

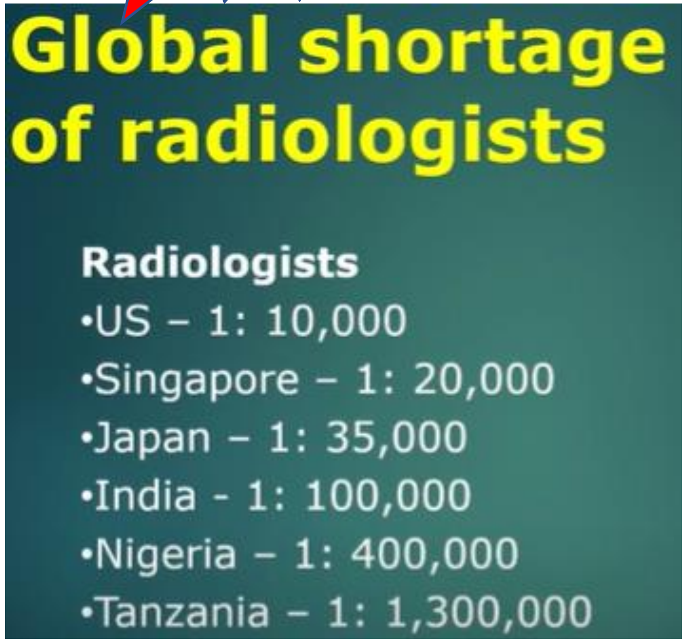
https://colab.research.google.com/drive/1-Kr_UwXgVdzj6zbXLG_UhYo6Z98HCoic?authuser=1#scrollTo=la7uXeQX8iOM

Why do we need COVID solution?

PRE COVID Statistics

Number of scans professionals have to analyze = 1,000 a day

This can lead to lengthy delays between scan and treatment – **even when someone needs urgent care**.



Global shortage of radiologists

Radiologists

- US – 1: 10,000
- Singapore – 1: 20,000
- Japan – 1: 35,000
- India - 1: 100,000
- Nigeria – 1: 400,000
- Tanzania – 1: 1,300,000

Solution Description

- Detecting COVID-19 with Chest X Ray using PyTorch Image classification of Chest X Rays in one of three classes:

  - *Normal,*
  - *Viral Pneumonia,*
  - *COVID-19*

- Dataset from COVID-19 Radiography Dataset on Kaggle

Youtube video

https://www.youtube.com/watch?v=07Kg3iGpzXA

Prevention of Car Burglary using Real Time Video Surveillance

sambath.narayanan@dataeverconsulting.com

Why do we need this solution ?

CCTV cameras and security staff are available in most of the residential/office complexes. Then why do we need this solution?

In any **large** complex, security is deployed in the selected locations.

Currently CCTV footage is used in majority of the cases for post-incident analysis

Hence an enhanced real time solution is required to address the above gaps.

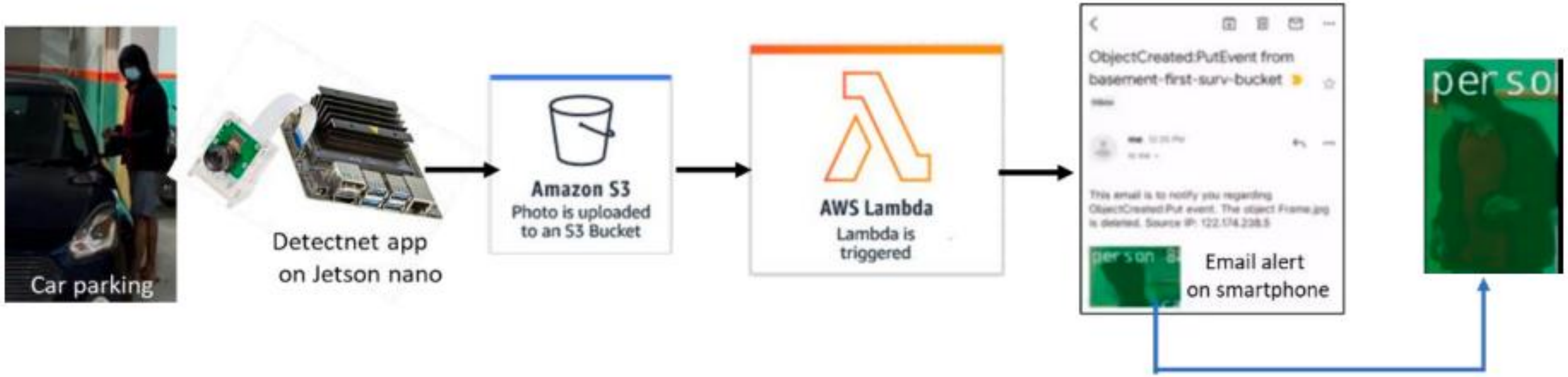
How the solution works ?

I. From the cctv video stream, Jetson Nano EDGE/AI based Object detector identifies a stranger moving around the car in odd hours and puts a bounding box

II. The application takes a snapshot and uploads it to AWS S3 storage

III. As soon as there is a put activity in S3 storage, Lambda function sends an email to the owners' smart phone with a photo of stranger

Solution components


Car parking
Detectnet app on Jetson nano
Amazon S3
Photo is uploaded to an S3 Bucket
AWS Lambda
Lambda is triggered
Email alert on smartphone
person
person

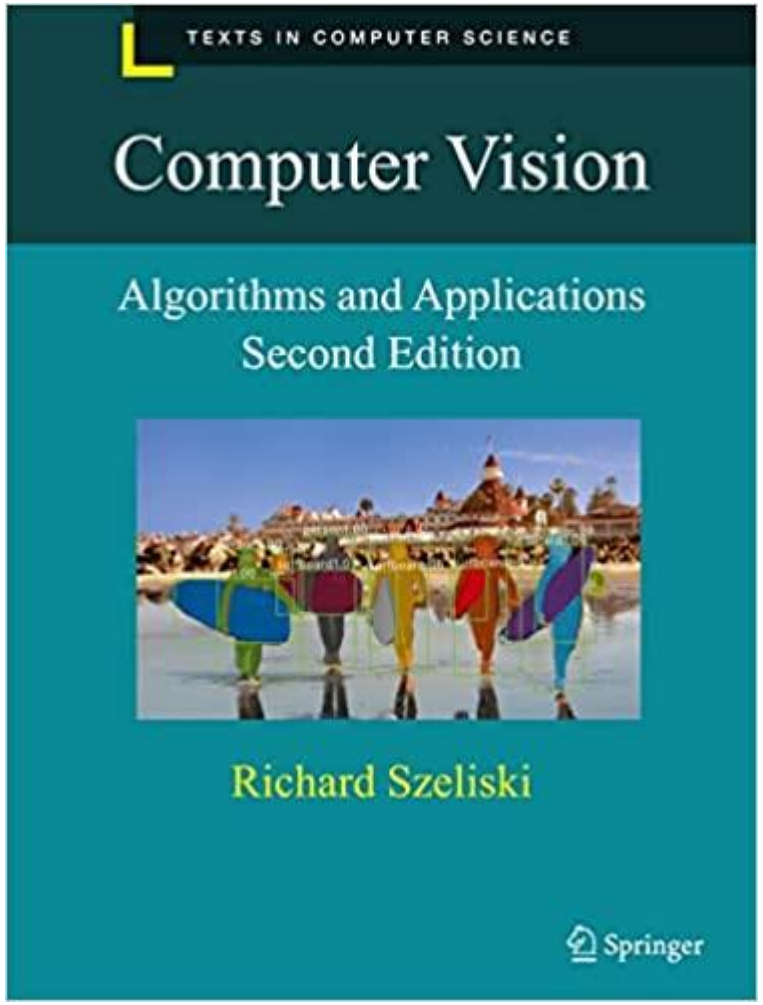
ObjectCreated:PutEvent from basement-first-surv-bucket

This email is to notify you regarding ObjectCreated:Put event. The object Frame.jpg is deleted. Source IP: 122.174.238.5

- Jetson Nano EDGE device
- Jetson-inference optimized library
- TRT

- cuDNN
- AWS S3, Lambda, SES
- Email on Smart Phone

Reference Book



TEXTS IN COMPUTER SCIENCE

Computer Vision

Algorithms and Applications
Second Edition

Richard Szeliski

Springer

Thank you !

Computer Vision

9th May 2023, Session 2B

sambath.narayanan@dataeverconsulting.com