TECH CONFERENCE
DotNet
2020

#DotNet2020

Rocket your Machine Learning models to the Edge with C#

ORGANIZATION

plain concepts

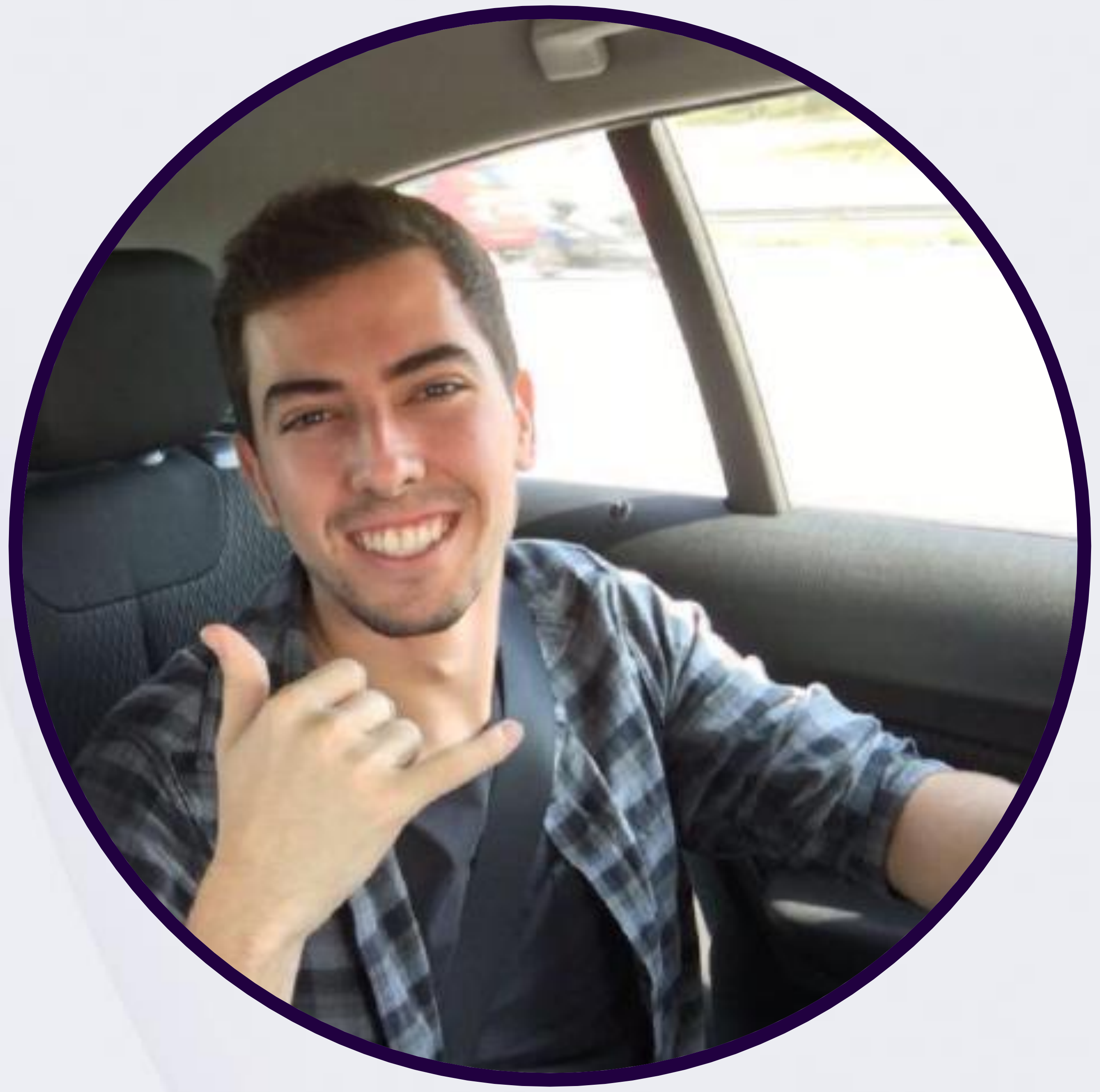
PLATINUM SPONSORS

Microsoft

COLLABORATORS

CEU | Universidad San Pablo

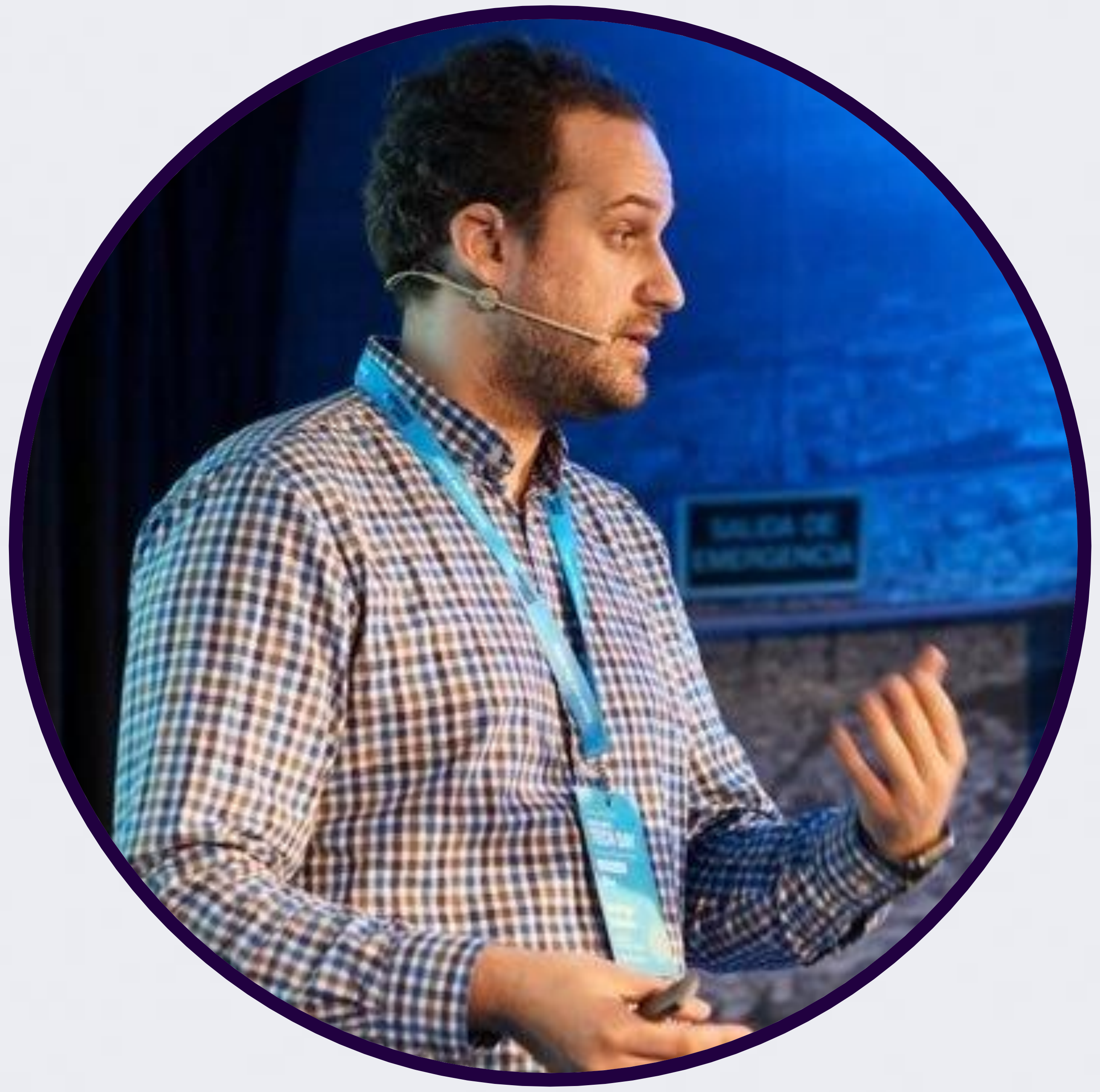
Thank you!



Alexander

AI Software Engineer

@alexndrglez
alexglezglez96@gmail.com



Rodrigo

AI Technical Lead

@mrcabellom
mrcabellom@gmail.com

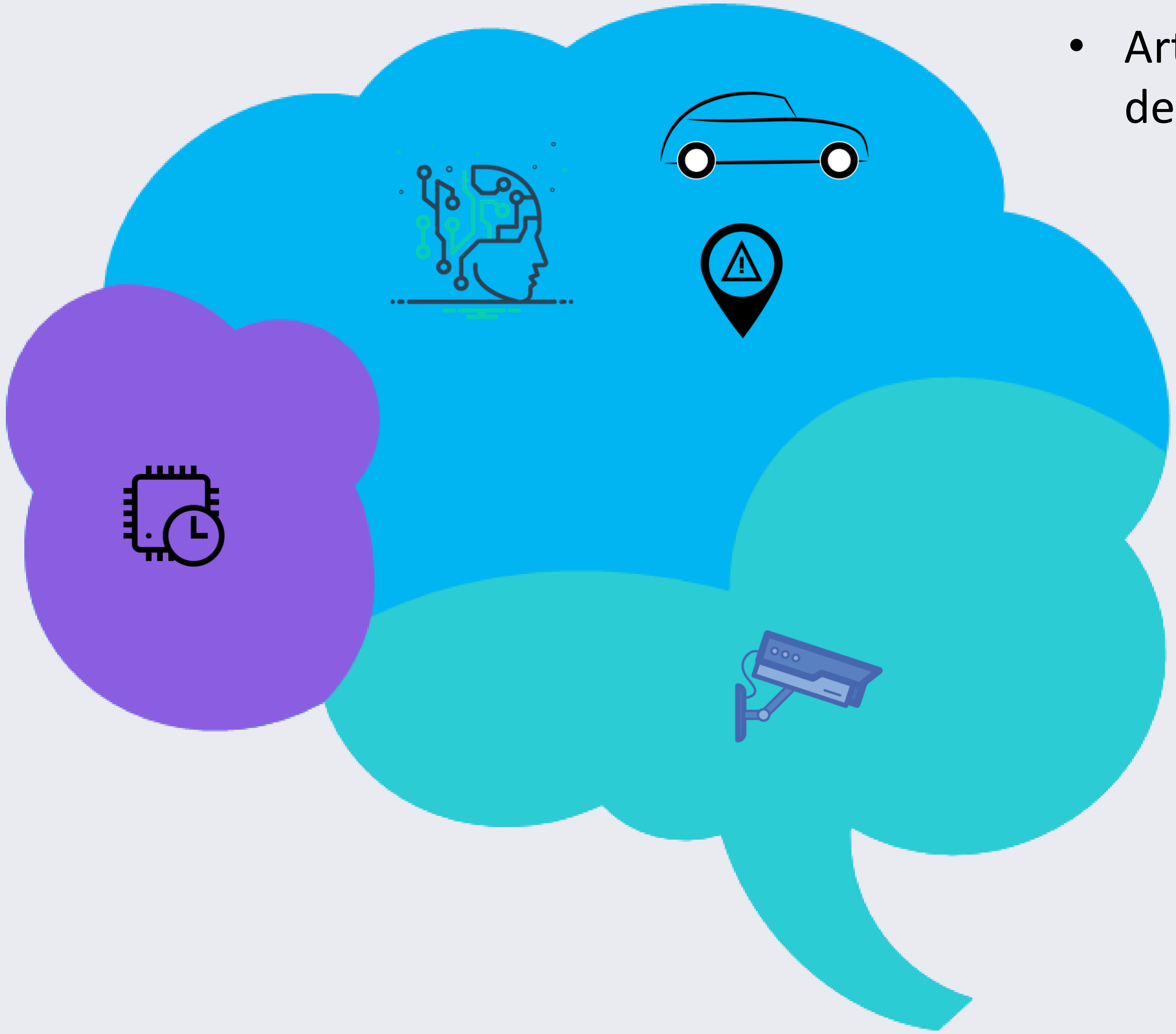
How can I build intelligent systems?

A neighborhood have a CCTV system to improve the security of the people who lives there. All the residents want to analyze the traffic inside the residential area in order to make it a safe place where their childrens can play outside.

How can I build intelligent systems?

Main challenges

- Artificial Intelligence techniques for object detection and tracking.
    - Car detection.
    - Danger zones detection.

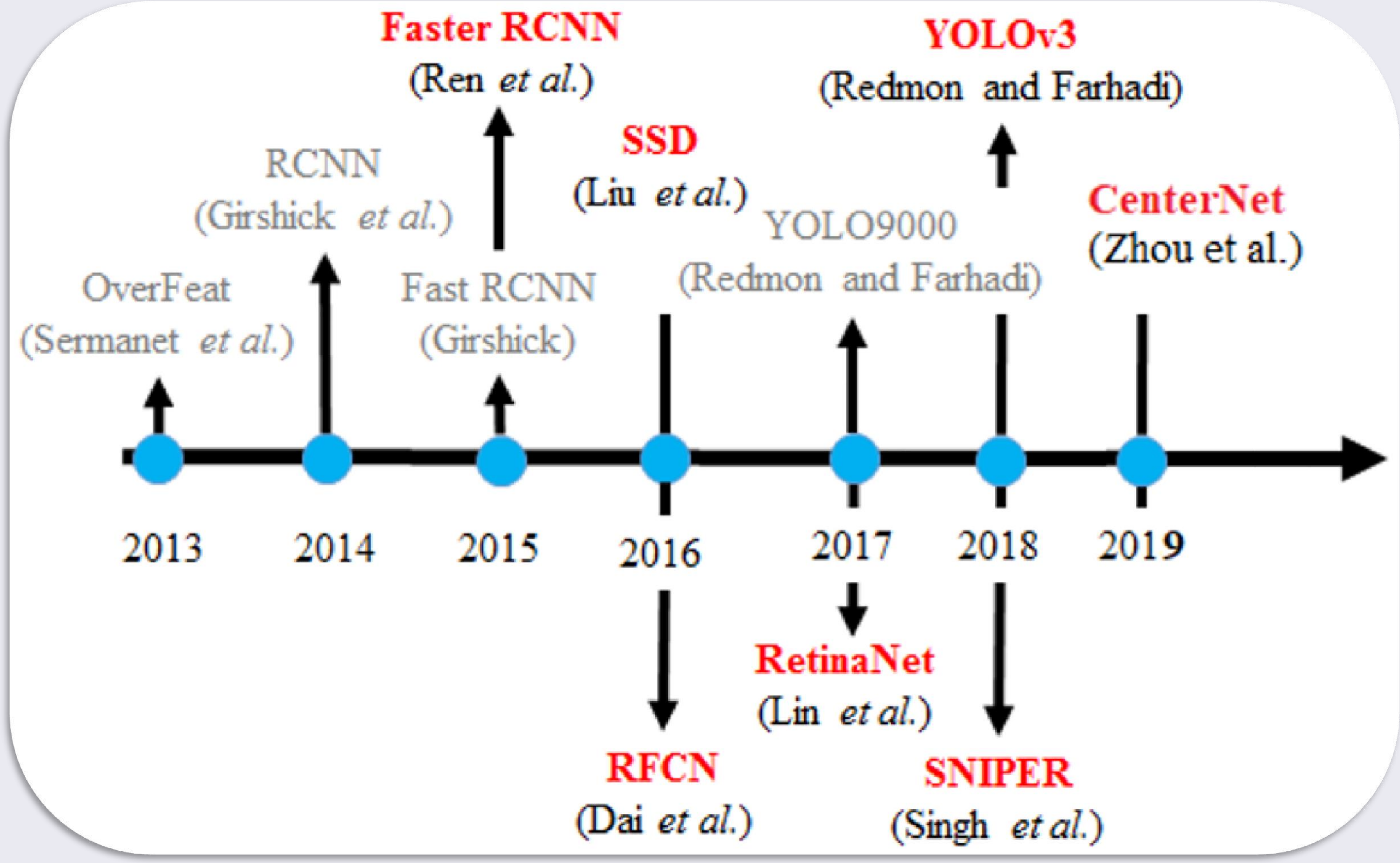


- Is our deep learning model ready to real-time inference?

- Do we have a real-time architecture able to work with a CCTV or a video management system?

Current status of Object detection - Tesla AutoPilot

Current status ML Object detection



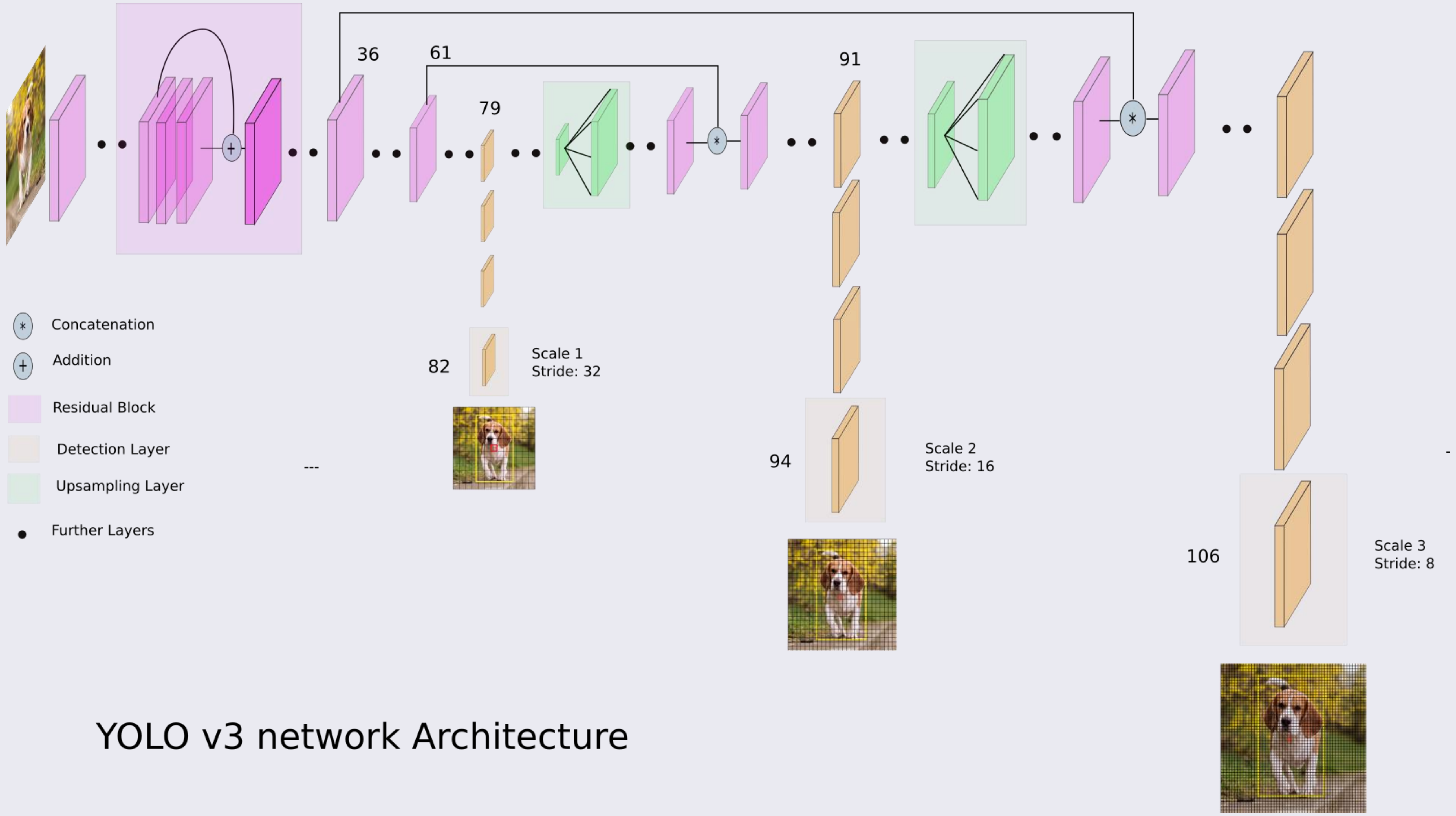
YoloV4
2020

YoloV5
2020

PP-Yolo
2020

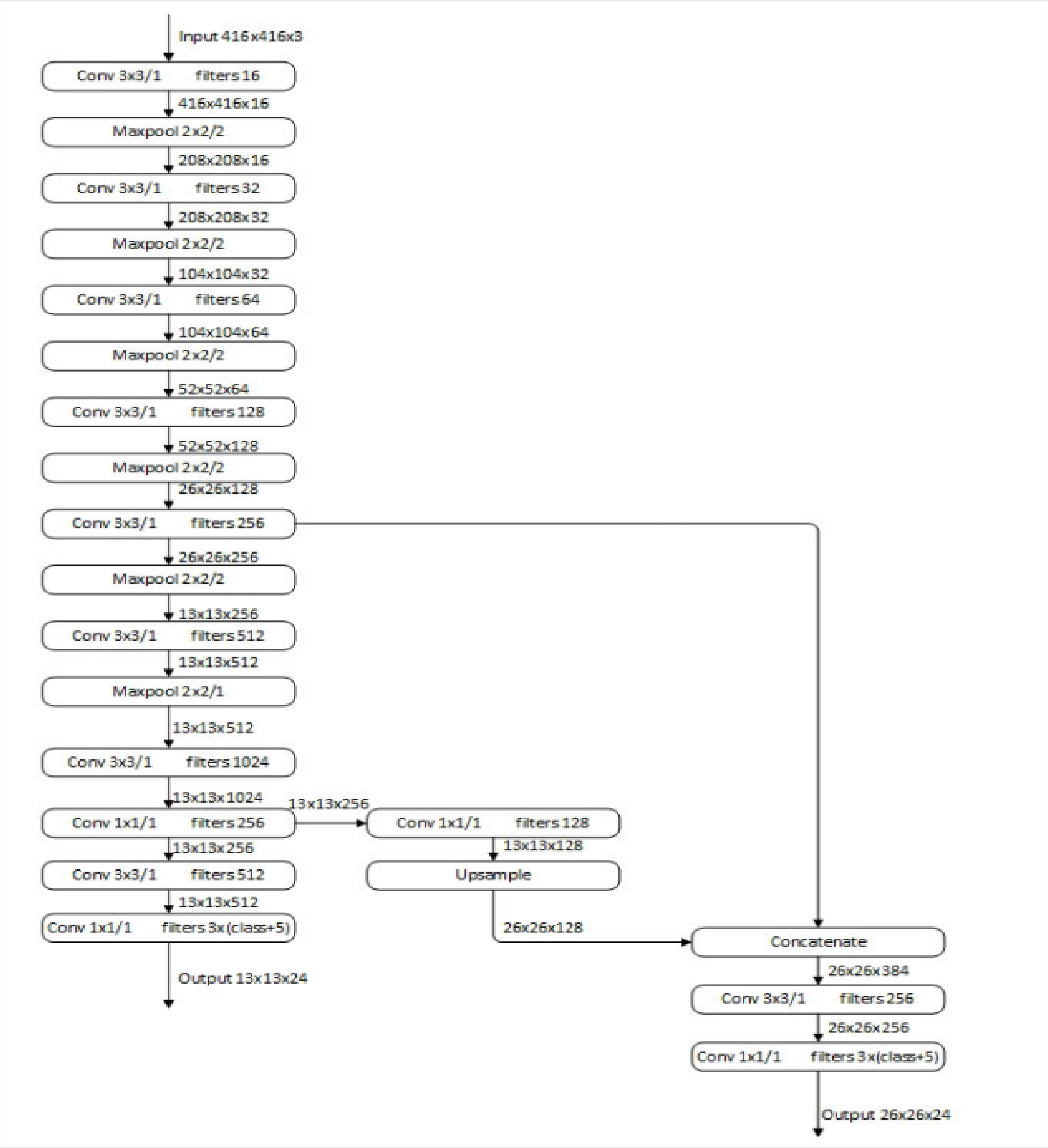
Current status ML Object detection – Yolo Models

YOLOv3



YOLO v3 network Architecture

Tiny YOLOv3



Real time object detection

Object detection in real-time video is considered to be much harder than image classification

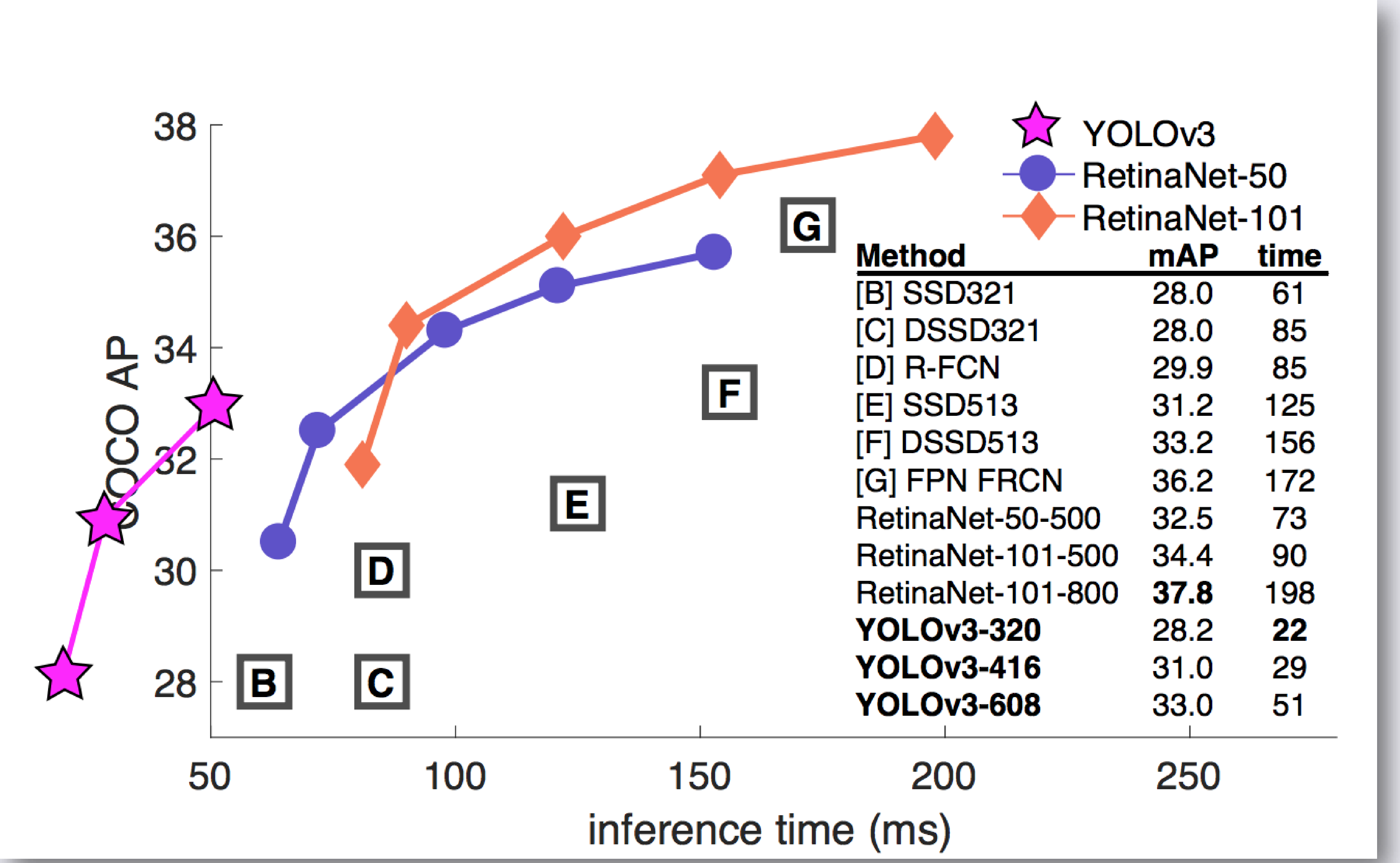
Key factors to consider:

- Neural network topology.
- Model compression
- Post processing optimization.
- Performance.

Real time object detection

Neural network topology

How to select a good network topology for real-time object detection?



Method	mAP	time
[B] SSD321	28.0	61
[C] DSSD321	28.0	85
[D] R-FCN	29.9	85
[E] SSD513	31.2	125
[F] DSSD513	33.2	156
[G] FPN FRCN	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	**37.8**	198
YOLOv3-320	28.2	**22**
YOLOv3-416	31.0	29
YOLOv3-608	33.0	51

- Data augmentation
- Training datataset
- Input image resolution
- Boundary box encoding
- Use of multi-scale images in training or testing
- ....

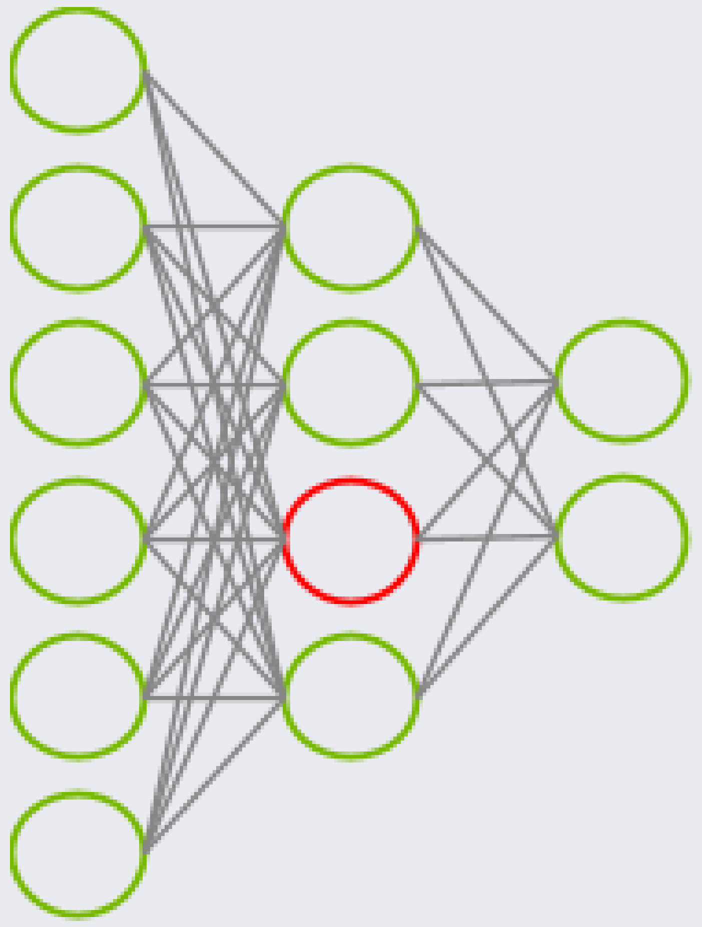
Real time object detection

Model compression

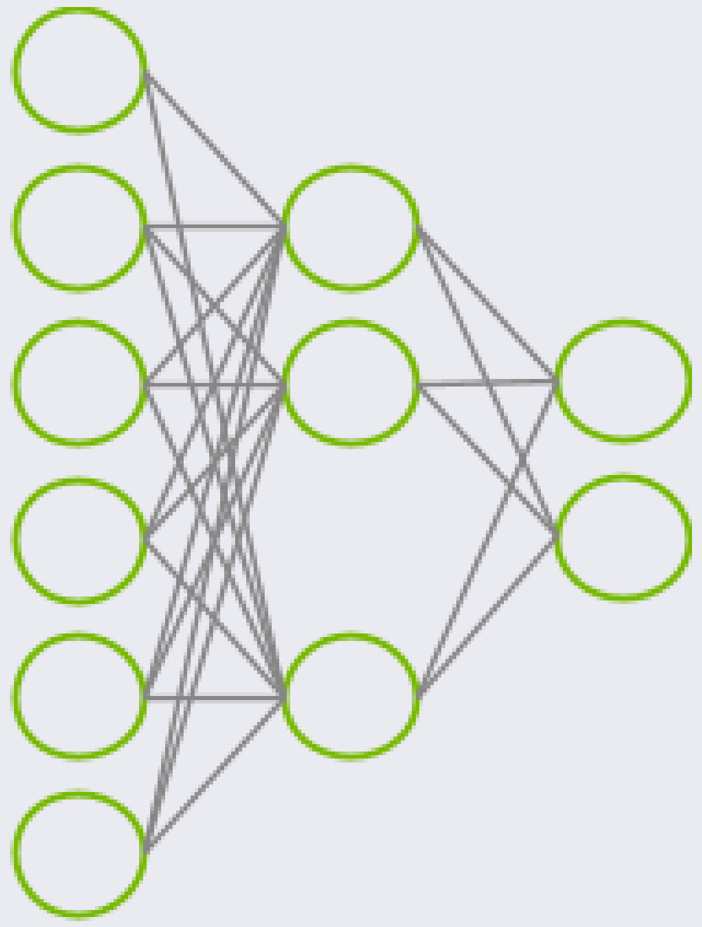
The goal of model compression is to achieve a model that is simplified from the original without significantly diminished accuracy (size/latency).

Compression techniques:

- Pruning.
- Quantization.
- Low-rank approximation and sparsity.
- Knowledge distillation.
- Neural Architectur Search (NAS).



6 inputs, 6 neurons (including 2 outputs), 32 connections



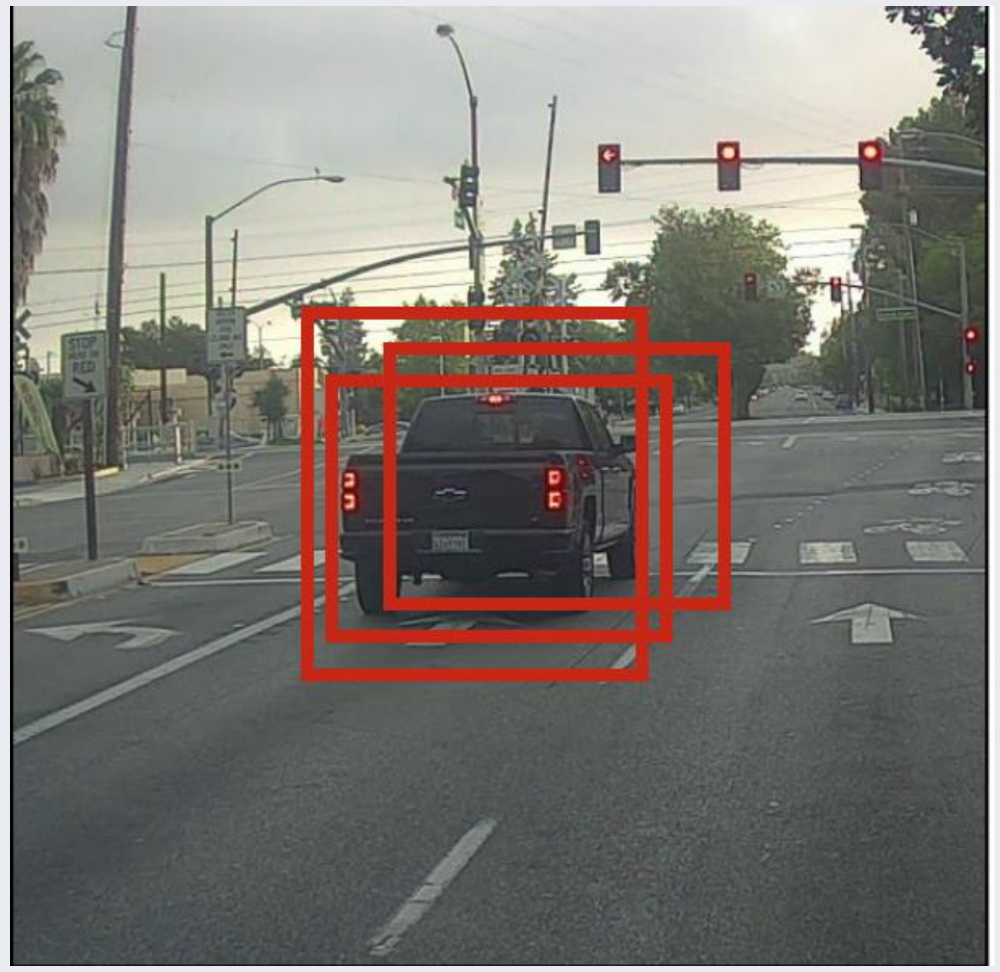
6 inputs, 5 neurons (including 2 outputs), 24 connections

Real time object detection

Post-processing optimization

The output of the Convolutional neural network is processed and converted into a form that can be fed to the non-max suppression (NMS).

Convert the processing pipeline in a pure vectorized format instead of relying on for-loops can increase the speed of our process

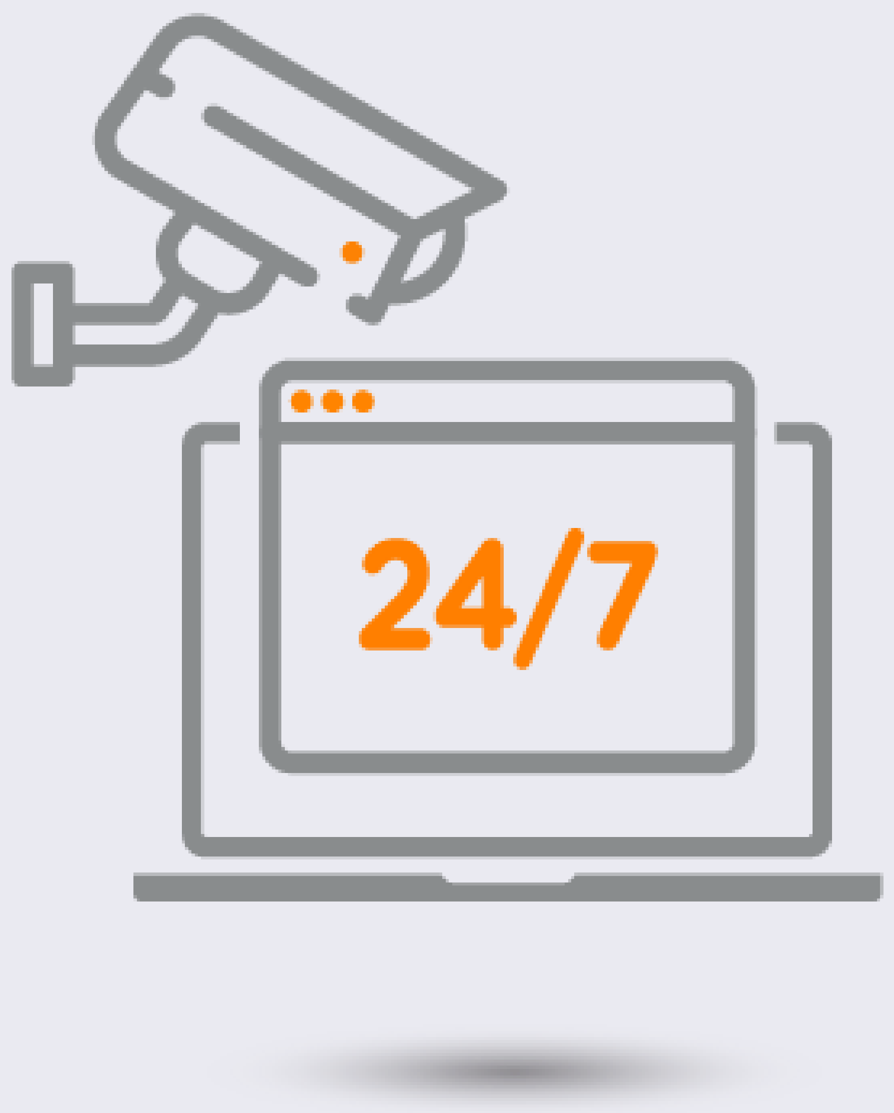


Real time object detection

Performance

The deployment of a real-time object detection system requires:

- How are we going to consume our model?
  - Achieve object detection with real-time throughput and low latency.
- Where is our model deployed?
  - Minimize the required computational resources allows more resources to be allocated for other tasks.



ML.NET

ML.NET components

Developer friendly API for Machine Learning

ML Model Training & Consumption

Data

- IDataView (Data set)
- File Loaders
- Database Loader
- Image Loader
- Data Transforms

Training ML tasks

- Classical ML
  - Classification
  - Regression
  - Anomaly Detection
  - Recommendations
  - Time Series
  - Ranking
  - Clustering
- Computer Vision
  - Image classification
  - Object Detection (*)

(40 trainers/algorithms)

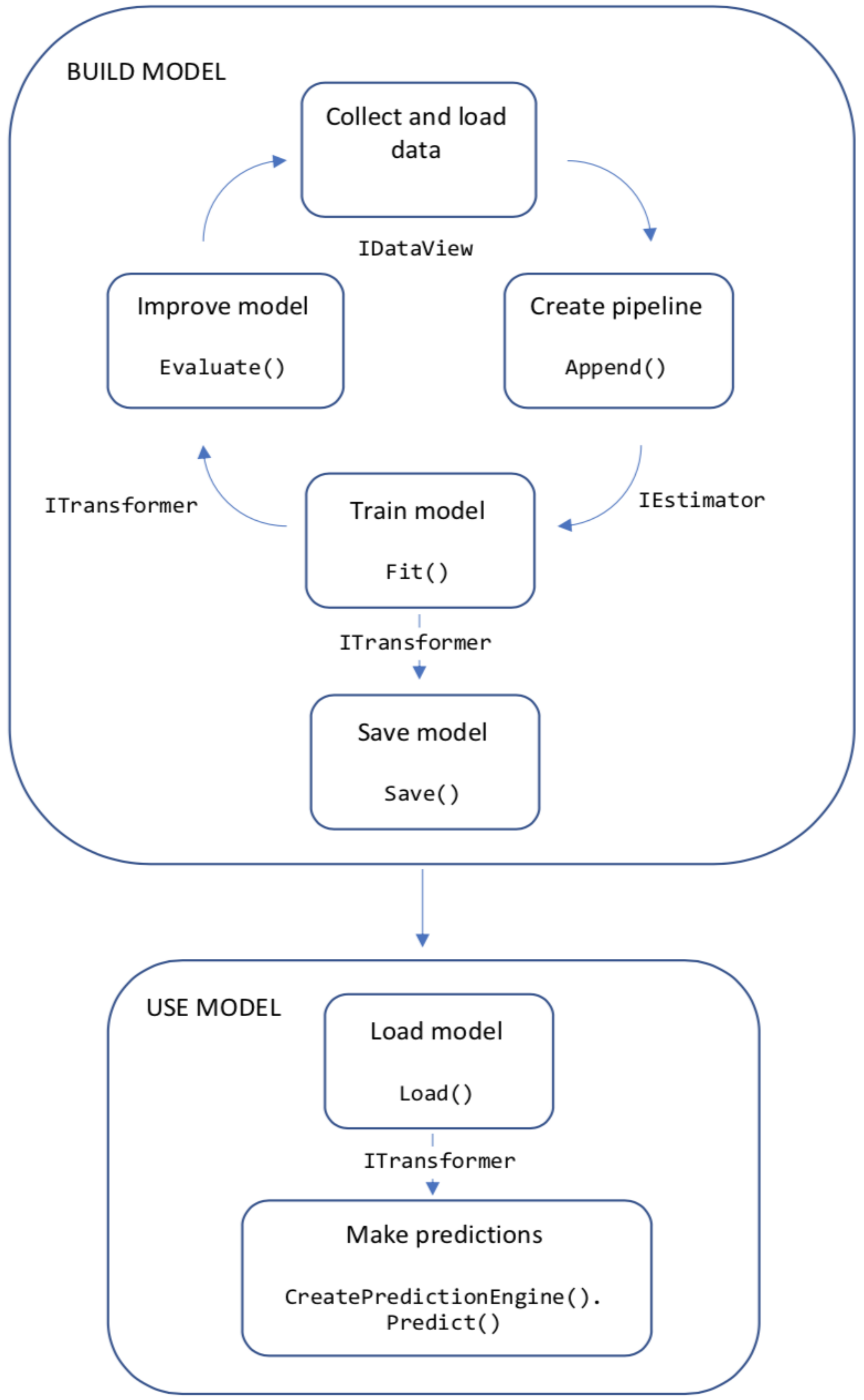
Model Consumption & Evaluation

- Consumption
  - Model
  - Prediction Engine
  - Prediction Engine Pool
- Evaluation
  - Model Evaluators
  - Quality metrics

Extensions & Tools

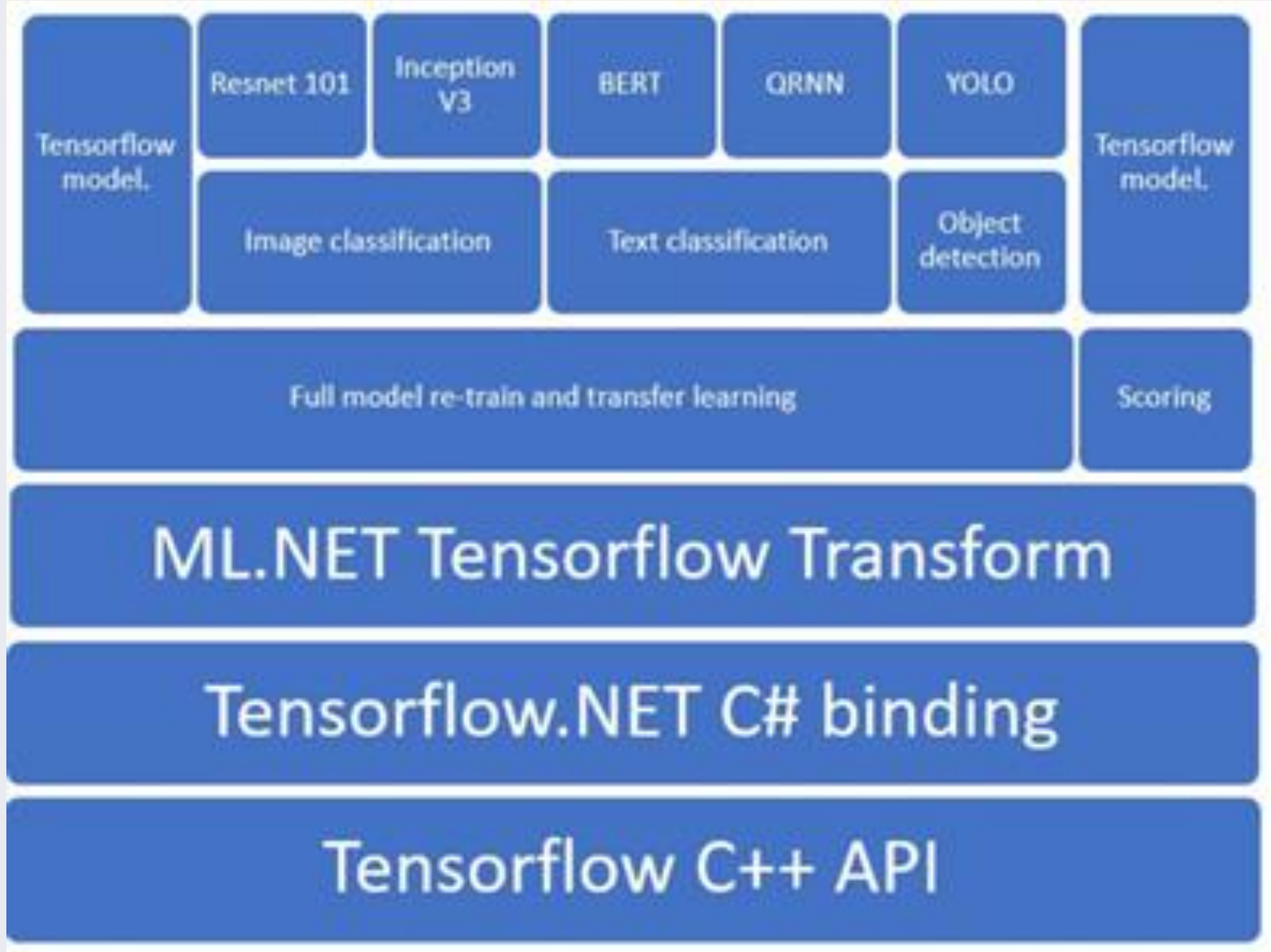
- ONNX consumption
- TensorFlow consumption
- AutoML
- CLI (Command-Line Interface)
- Model Builder in VS

(*) Object detection coming soon after v1.4-Preview



ML.NET 19-20 Updates:

- Image classification based on deep neural network retraining with GPU support
- Improvements in for image classification and object detection (Tensorflow.NET library)
- GPU support on Windows and Linux
- Added additional supported DNN architectures to the Image Classifier
  - Inception V3
  - ResNet V2 101
  - Resnet V2 50
  - Mobilenet V2

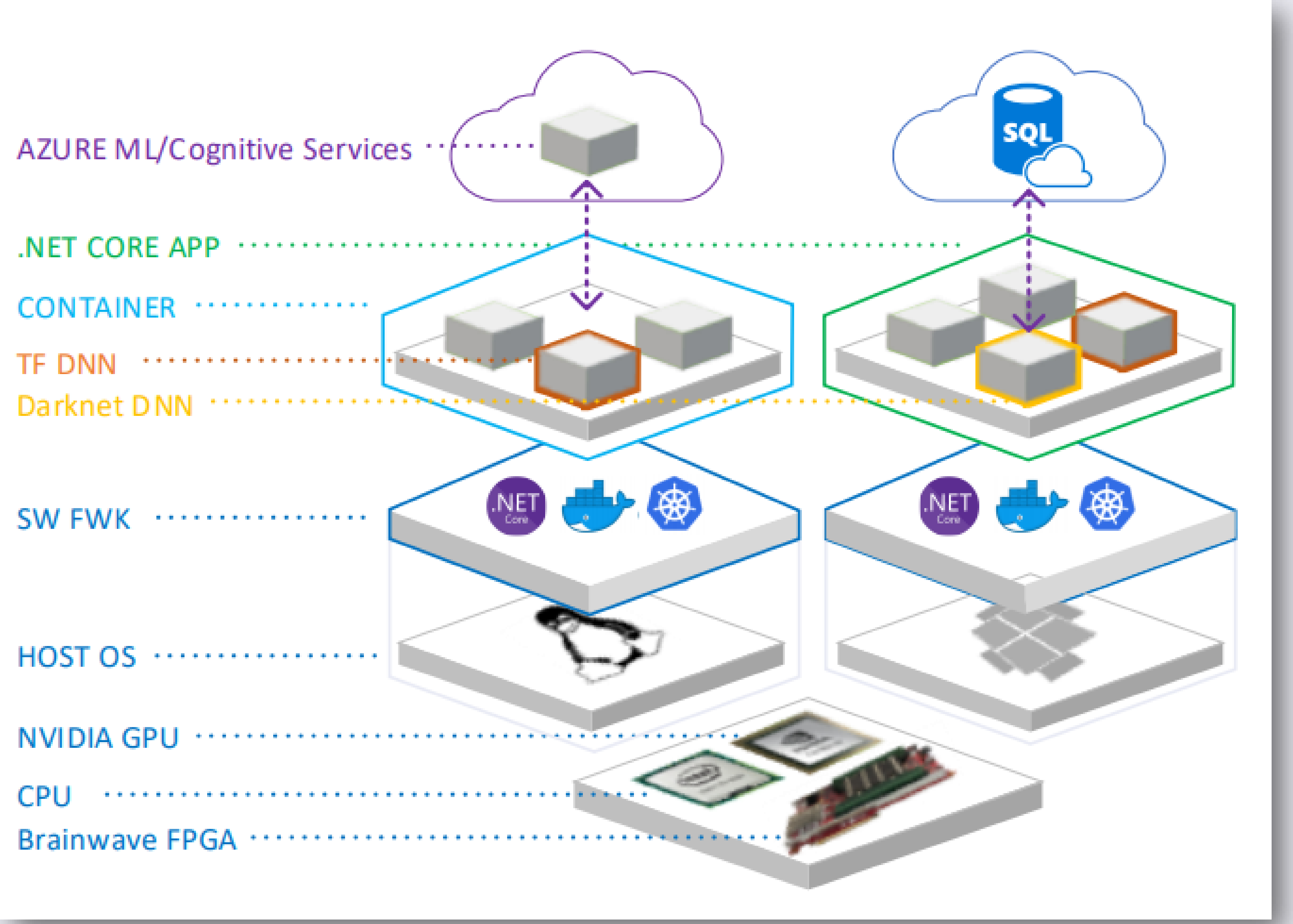


Project Rocket

A powerful configurable platform for live video analytics

Project Rocket's goal is to *democratize* video analytics: build a system for real-time, low-cost, accurate analysis of live videos.

- Built on C# .Net Core
- Plug any deep learning model: Tensorflow, Darknet, Onnx
- Custom models support
- Simpler motion filters (OpenCV)
- GPU/FPGA Acceleration
- Docker containerization



Project Rocket

Pipelines

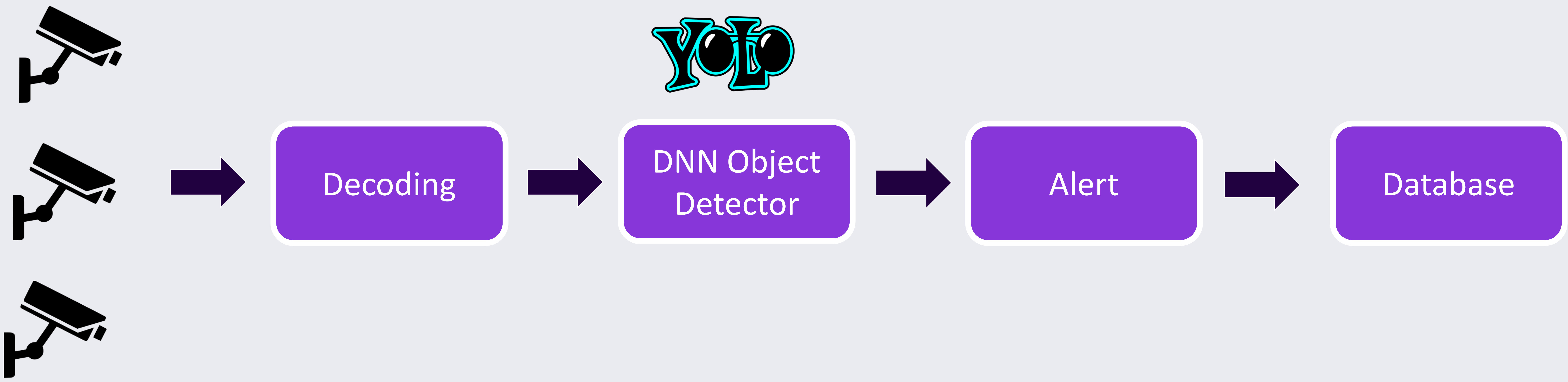
Five pre-built video analytics pipelines

1. **Alerting on objects (Darknet Yolo V3)**
2. Alerting on objects (Fast R-CNN)
3. Detecting objects with cascaded DNNs cheap filters, and after-the-fact querying
4. Detecting objects
5. Edge/Cloud split (Azure Machine Learning)
6. Edge/Cloud split + containers

```
static void Main(string[] args)
{
    while (true)
    {
        //decoder
        Mat frame = decoder.getNextFrame();

        //background subtractor
        List<Box> foregroundBoxes = bgs.DetectObjects(DateTime.Now, frame, frameIndex, out fgmask);

        //line detector
        occupancy = lineDetector.updateLineOccupancy(frame, frameIndex, fgmask, foregroundBoxes);

        //cheap TensorFlow DNN

    }
}
```

Project Rocket

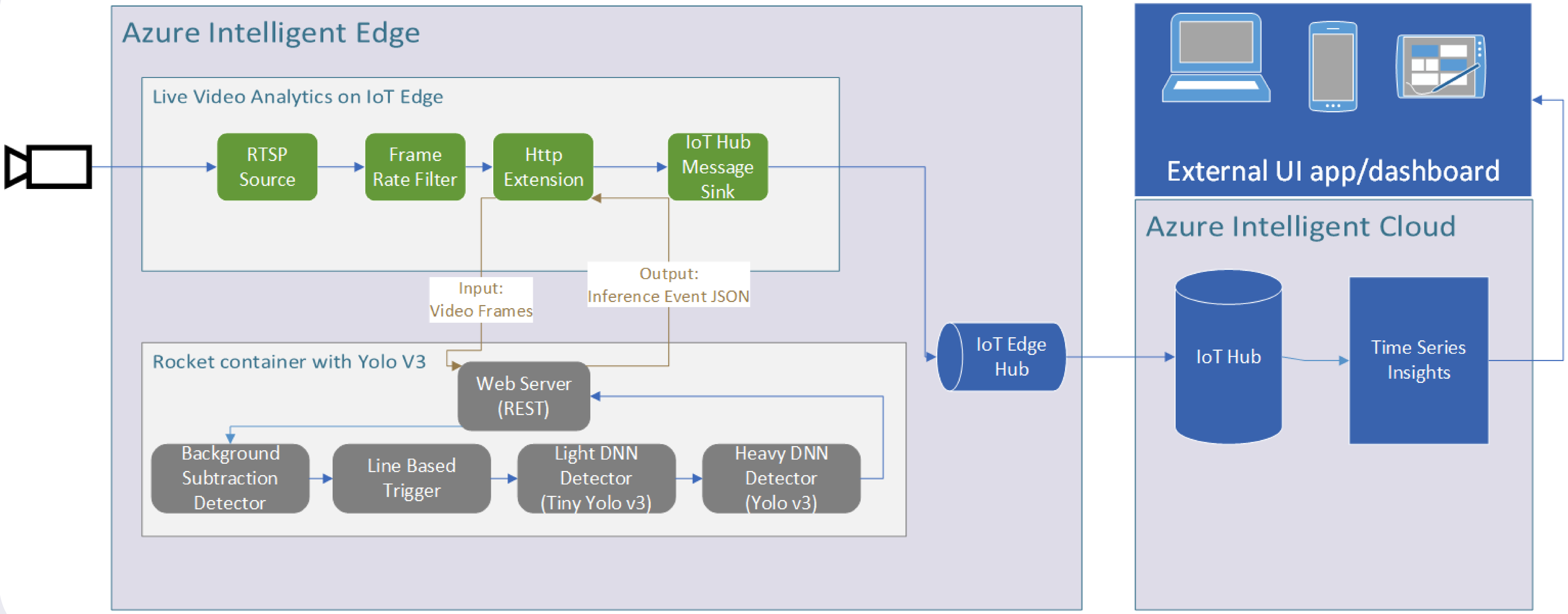
Pipelines



Demo

Microsoft Rocket Video Analytics Platform

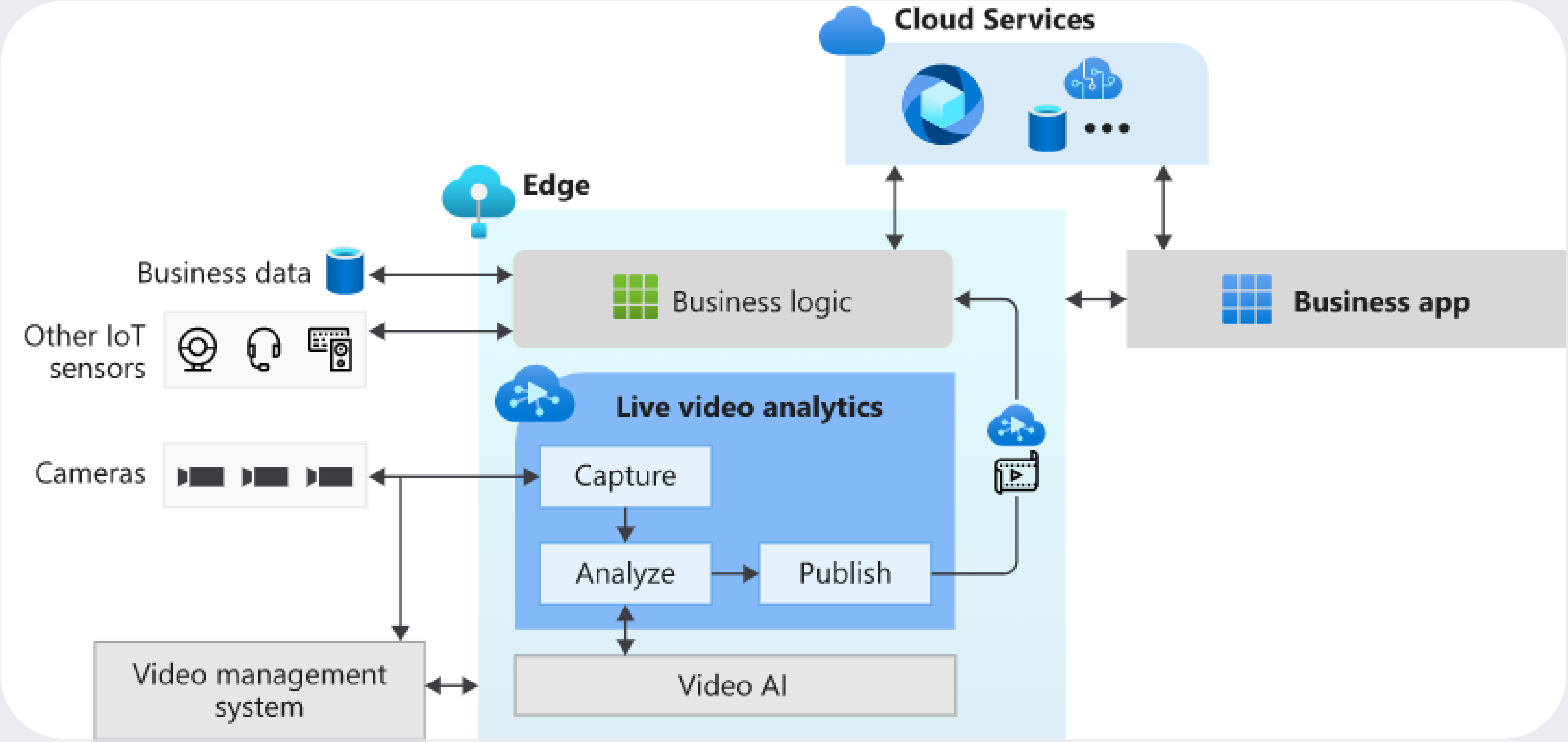
Arquitectura Live video analytics + Rocket



Live video analytics on IoT Edge

Key terms to undertand LVA:

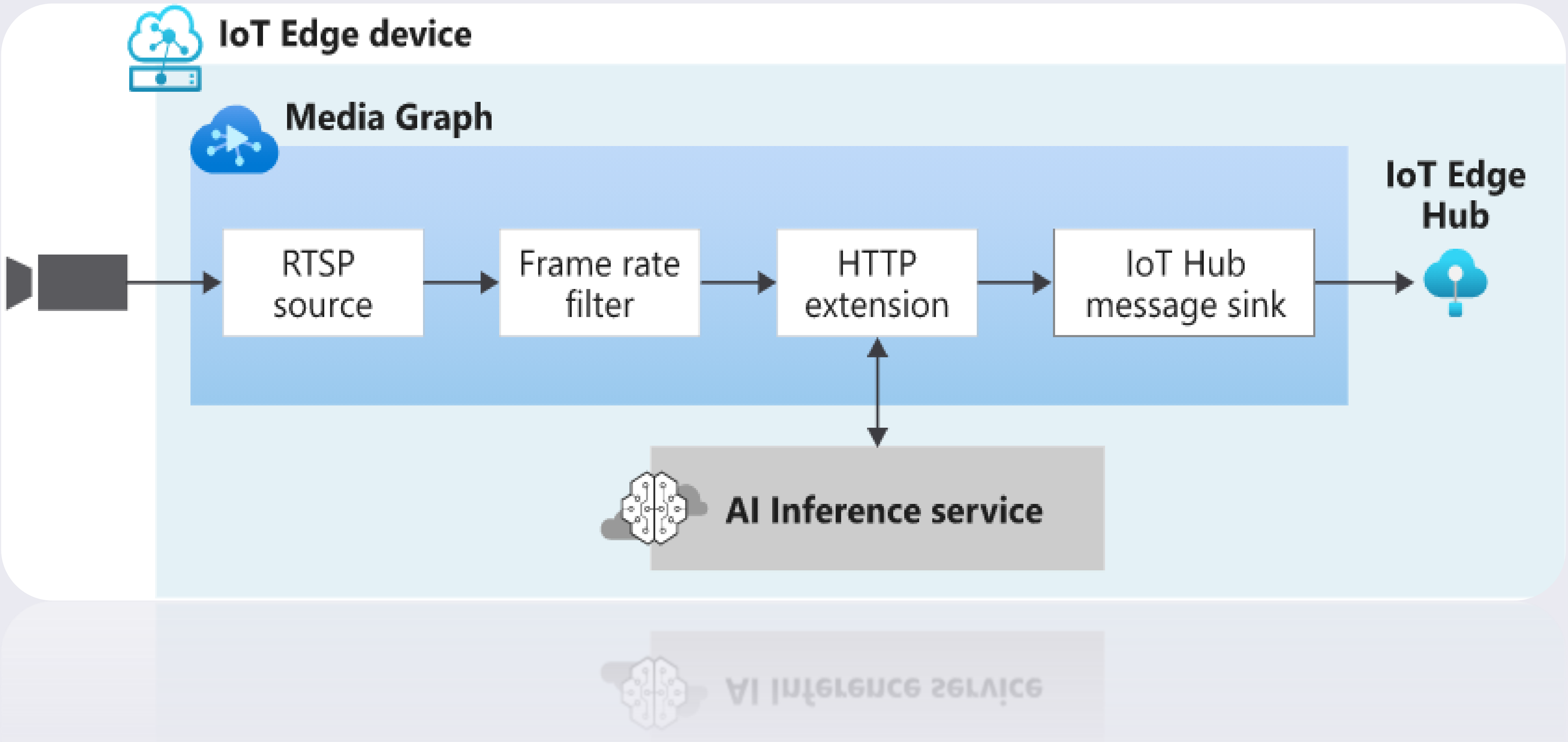
- Media graph
- Video recording
- Video playback
- Continuous video recording
- Event-based video recording
- Live Video Analytics without video recording



Live Video Analytics:
Analyze live video by using your own HTTP model

LVA steps in our demo:

1. An edge module simulates an IP camera hosting a Real-Time Streaming Protocol (RTSP) server.
2. An RTSP source node pulls the video feed from this server and sends video frames to the frame rate filter processor node.
3. This processor limits the frame rate of the video stream that reaches the HTTP extension processor node.
4. The HTTP extension node plays the role of a proxy. It converts the video frames to the specified image type. Then it relays the image over REST to another edge module that runs an AI model behind an HTTP endpoint.
5. AI Edge module is built by using the YOLOv3 model to detects objects
6. The HTTP extension processor node gathers the detection results and publishes events to the IoT Hub sink node. The node then sends those events to IoT Edge Hub.

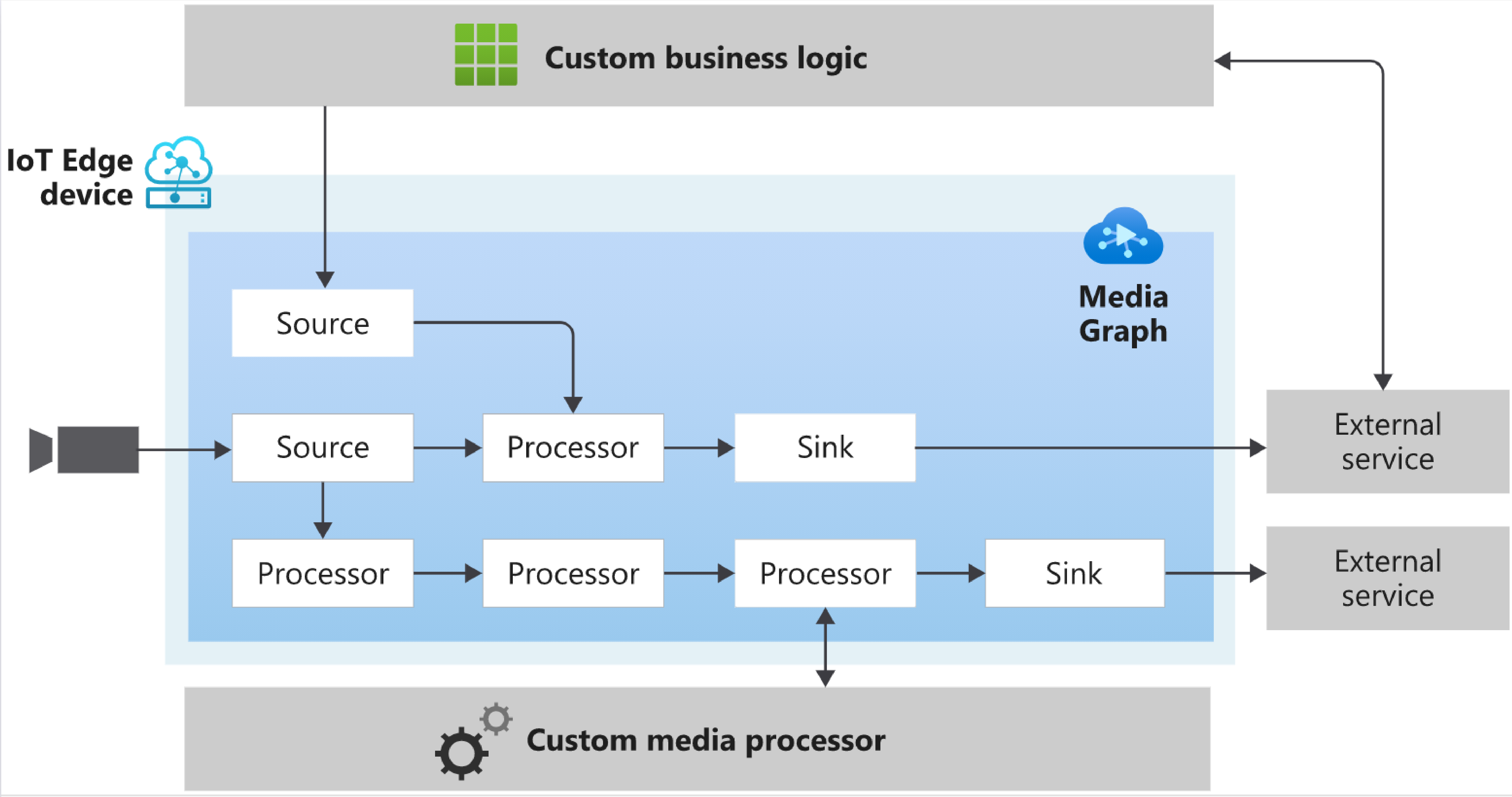


Video streaming (Media graph)

A media graph lets you define where media should be captured from, how it should be processed, and where the results should be delivered.

Live video analytics supports different types of nodes:

- Source nodes (RTSP, ONVIF)
- Processor nodes (Frame rate filter)
- Sink nodes (Iot Hub)

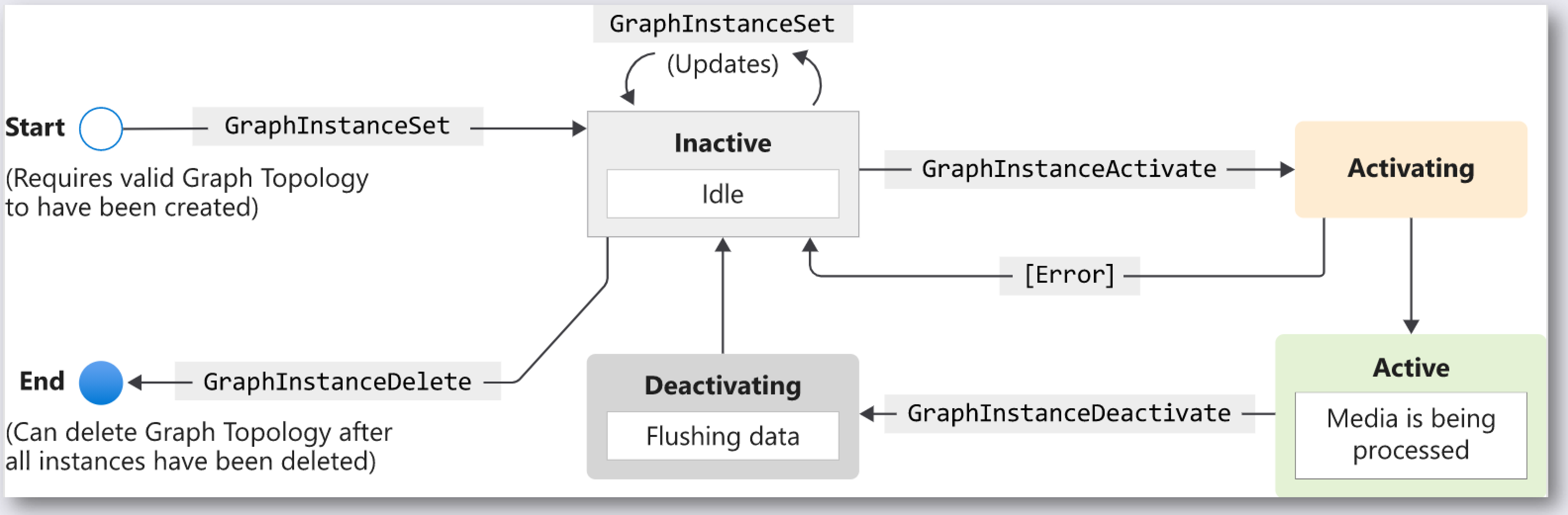


Media graph

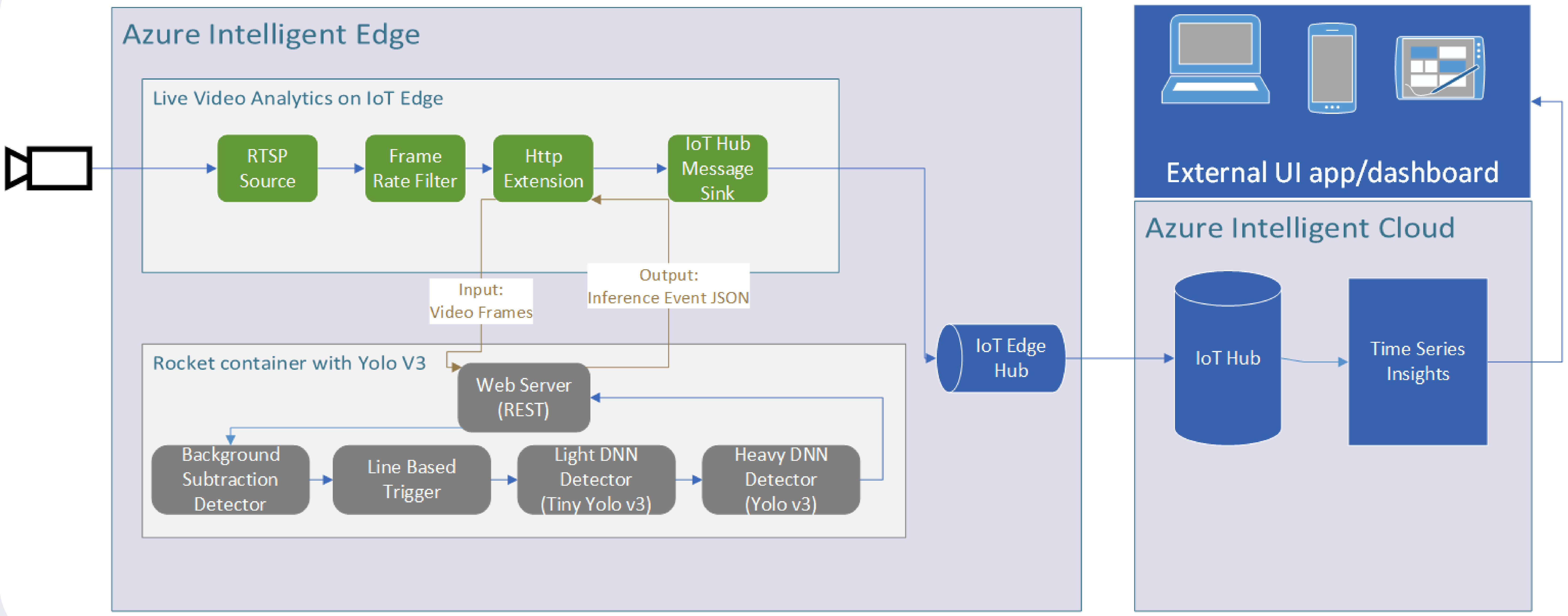
Topologies and instances

Live video analytics allow us to manage media graphs via two concepts:

- Graph topology
- Graph instance

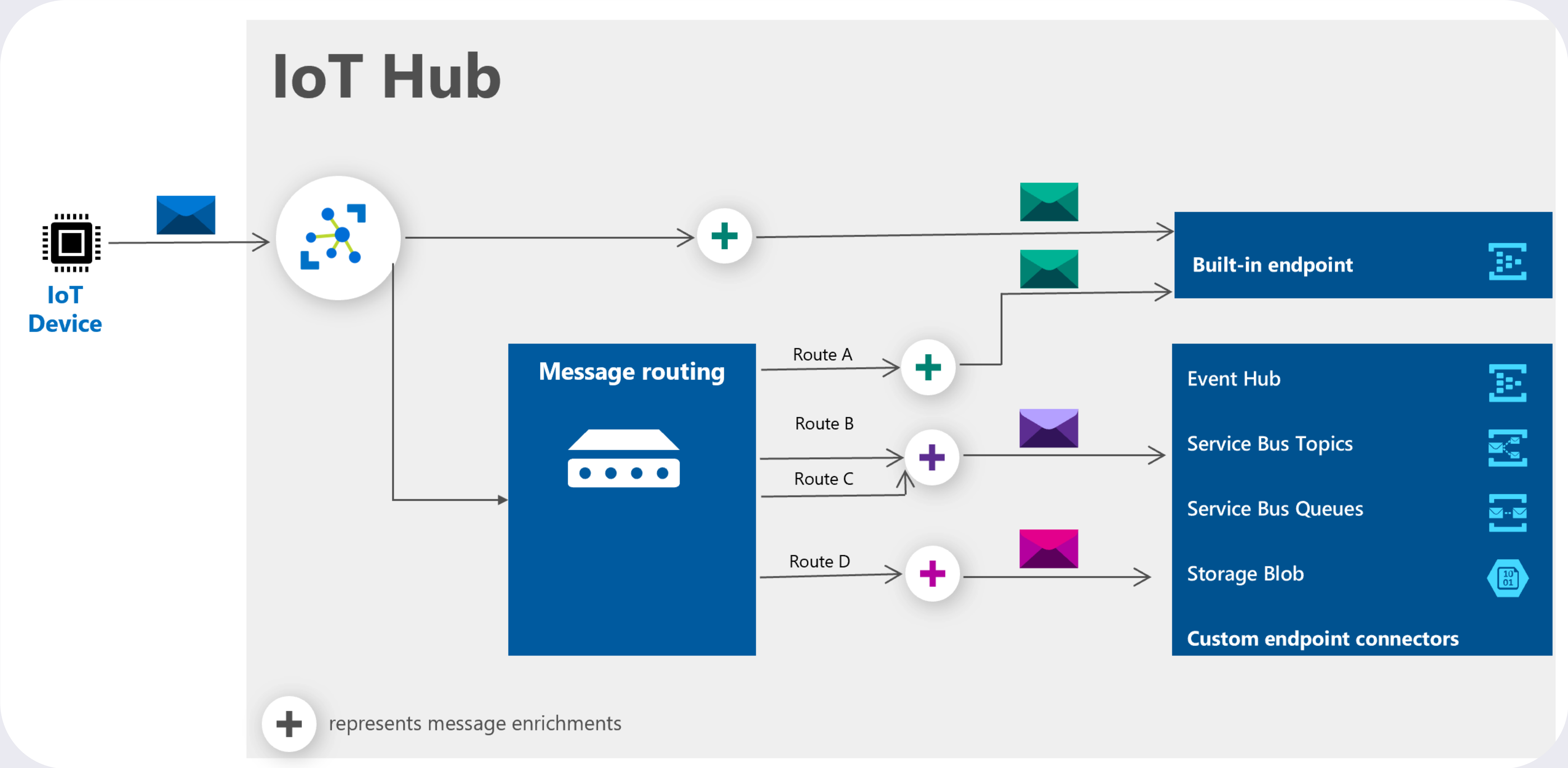


Azure IoT – Hub and Edge



Azure IoT Hub

- Scale your solution
- Secure your communications
- Route device data
- Integrate with other services
- Configure and control your devices
- Make your solution highly available
- Connect your devices



Azure IoT Edge

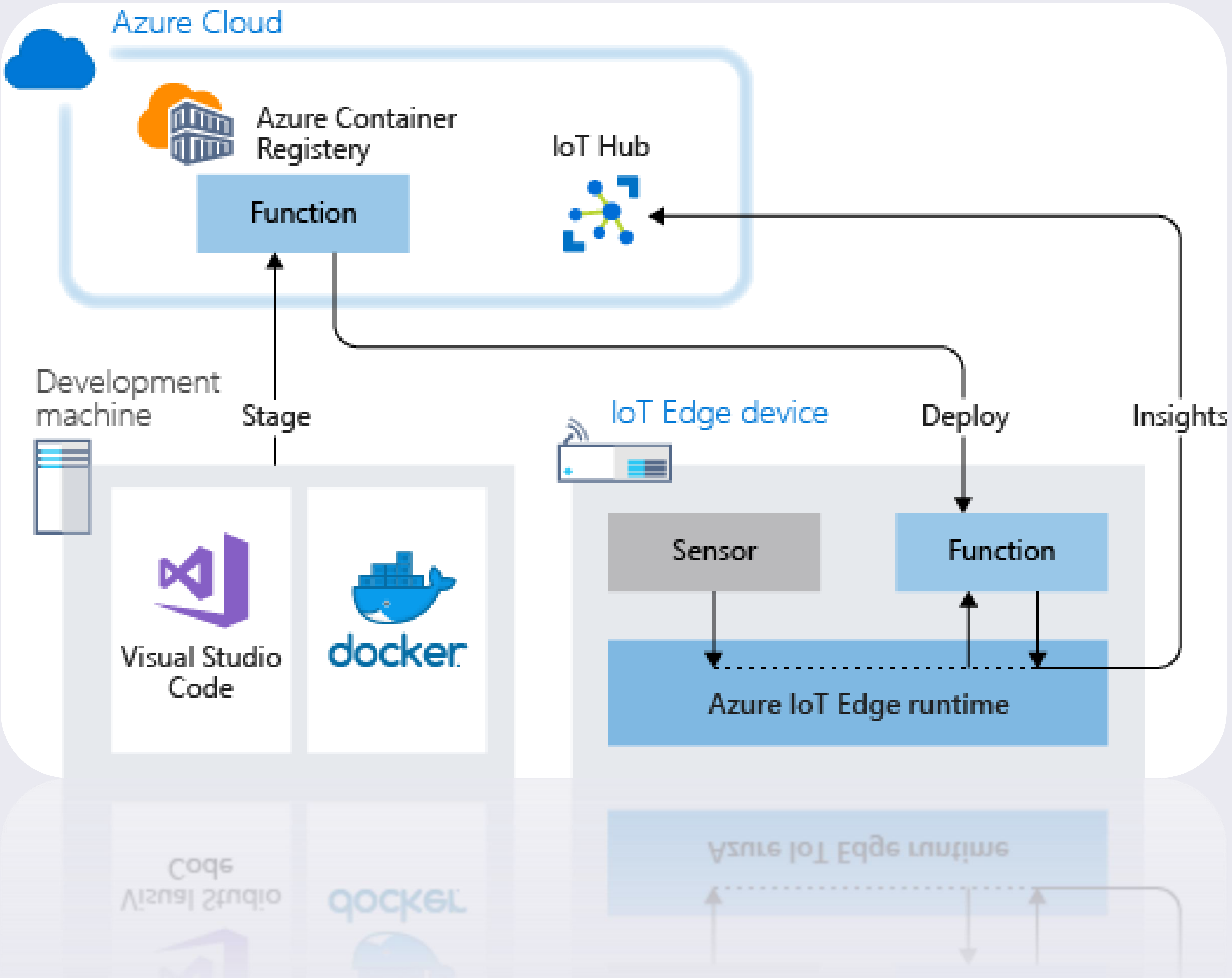
Azure IoT Edge is made up of three components:

• **IoT Edge modules** are containers that run Azure services, third-party services, or your own code. Modules are deployed to IoT Edge devices and execute locally on those devices.

• **The IoT Edge runtime** runs on each IoT Edge device and manages the modules deployed to each device.

1. Installs and update workloads on the device.
2. Maintains Azure IoT Edge security standards on the device.
3. Ensures that IoT Edge modules are always running.
4. Reports module health to the cloud for remote monitoring.
5. Manages communication between downstream leaf devices and an IoT Edge device, between modules on an IoT Edge device, and between an IoT Edge device and the cloud

• **A cloud-based interface** enables you to remotely monitor and manage IoT Edge devices.



Demo

Live Video Analytics with Microsoft Rocket

Questions & Answers

Thanks and ...
See you soon!

Thanks also to the sponsors.
Without whom this would not have been posible.

plain concepts

Microsoft

axazure

plasticscm

CEU | Universidad San Pablo