Distributed inference in IoT clusters

Dr. Victor Cionca
Munster Technological University

Bio

Lecturer in Computer Science @ MTU Cork

PhD in Wireless Sensor Networks from University of Limerick, 2012

Lead PI in MoNet project (MTU-TUTF funding).
FI in SFI CONFIRM Centre.
FI in SFI CONNECT Centre.
NI in SFI ADVANCE CRT.

High speed wireless networks.

Network control, optimization, management.

Programmable networks.

Today’s outline

The challenges of ML-based inference in IoT devices

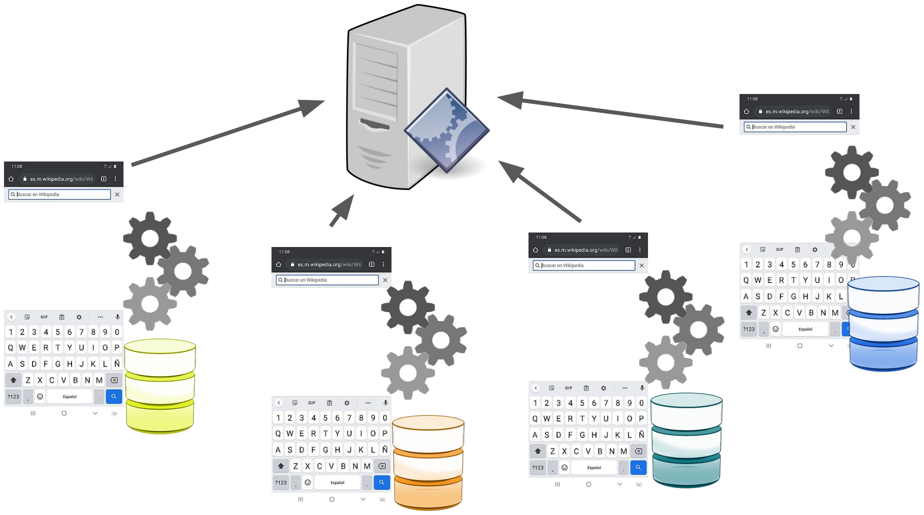
Distributing inference between IoT devices

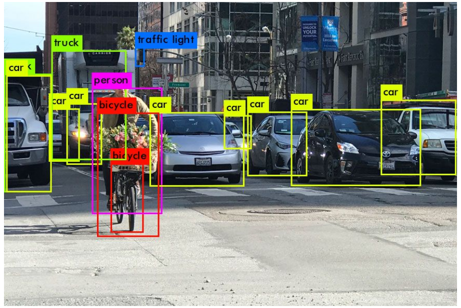
Communication challenges.

Machine Learning in IoT

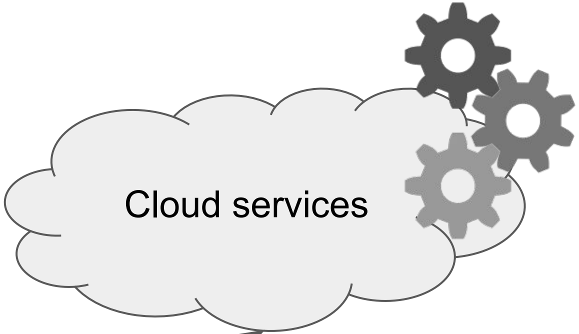
Federated Learning
- Train on local data (very slow)
- Aggregate model centrally.

Inference
- Far more common
- Still quite slow
- 2017 flagship smartphone ~100s for image classification. (Huynh et al, “Deepmon…”, ACM MobiSys 2017)

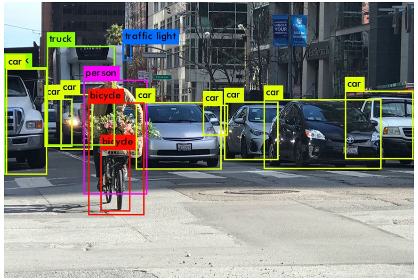




1 Acquire data



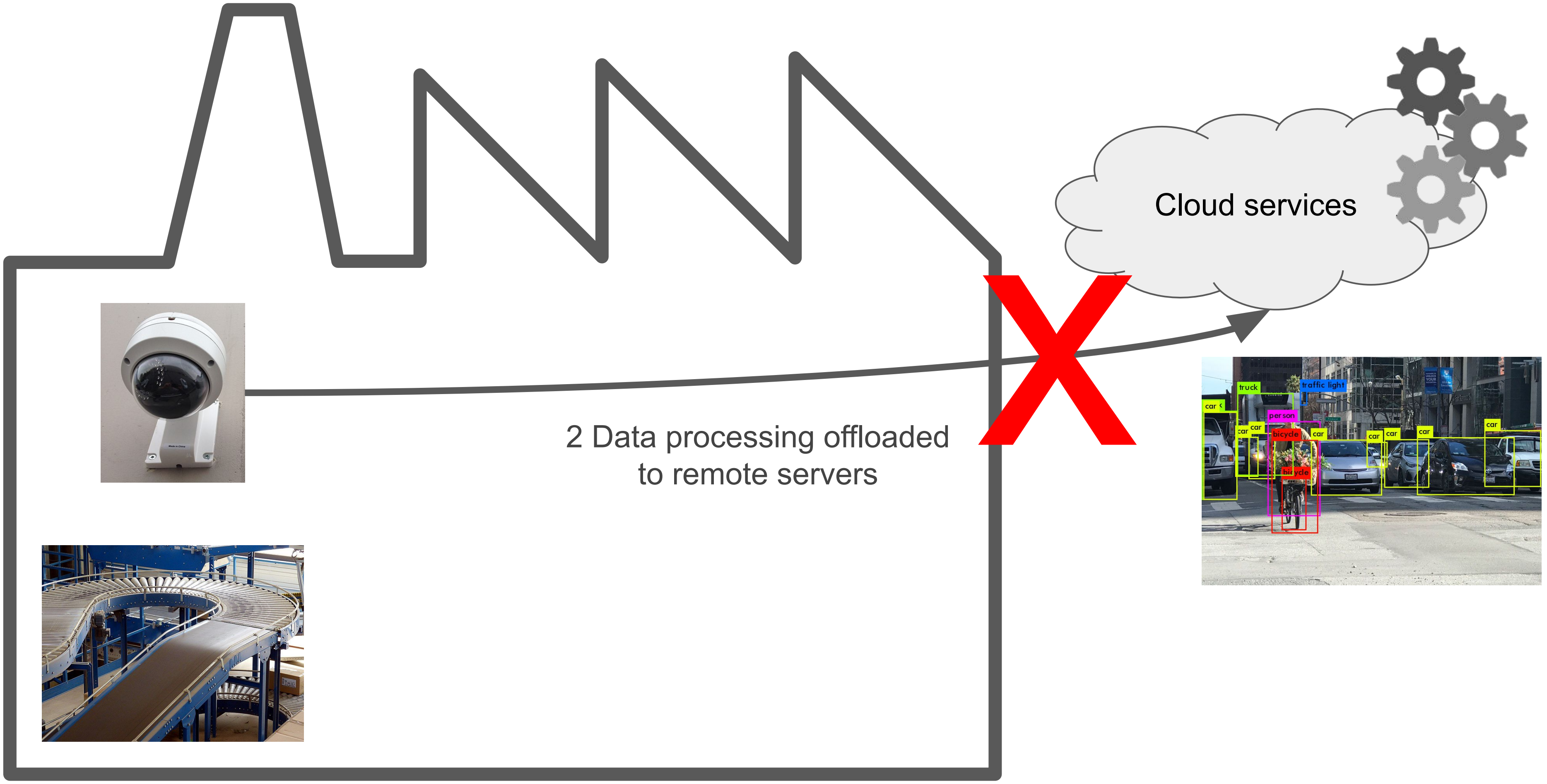
2 Data processing offloaded
to remote servers

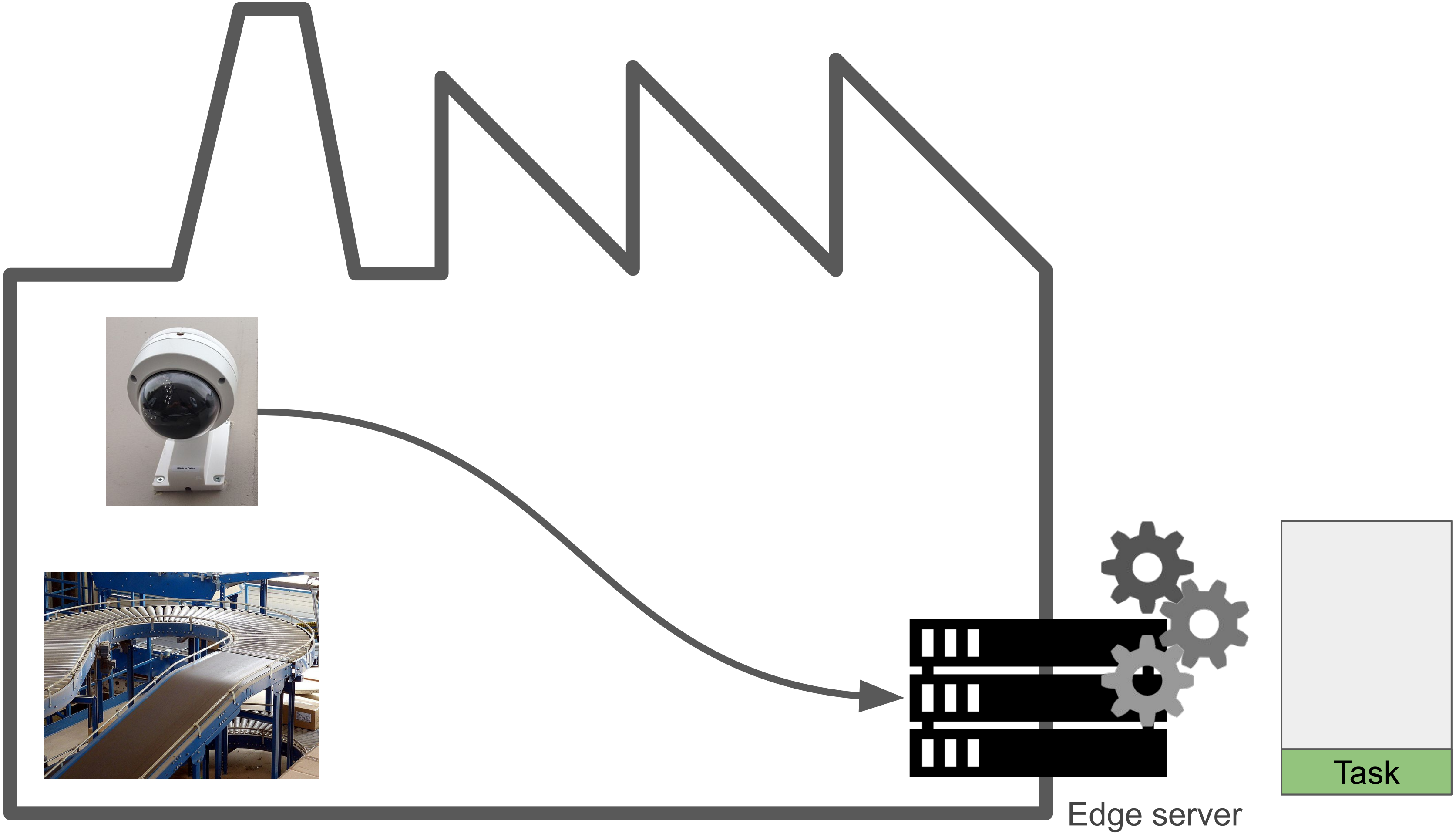


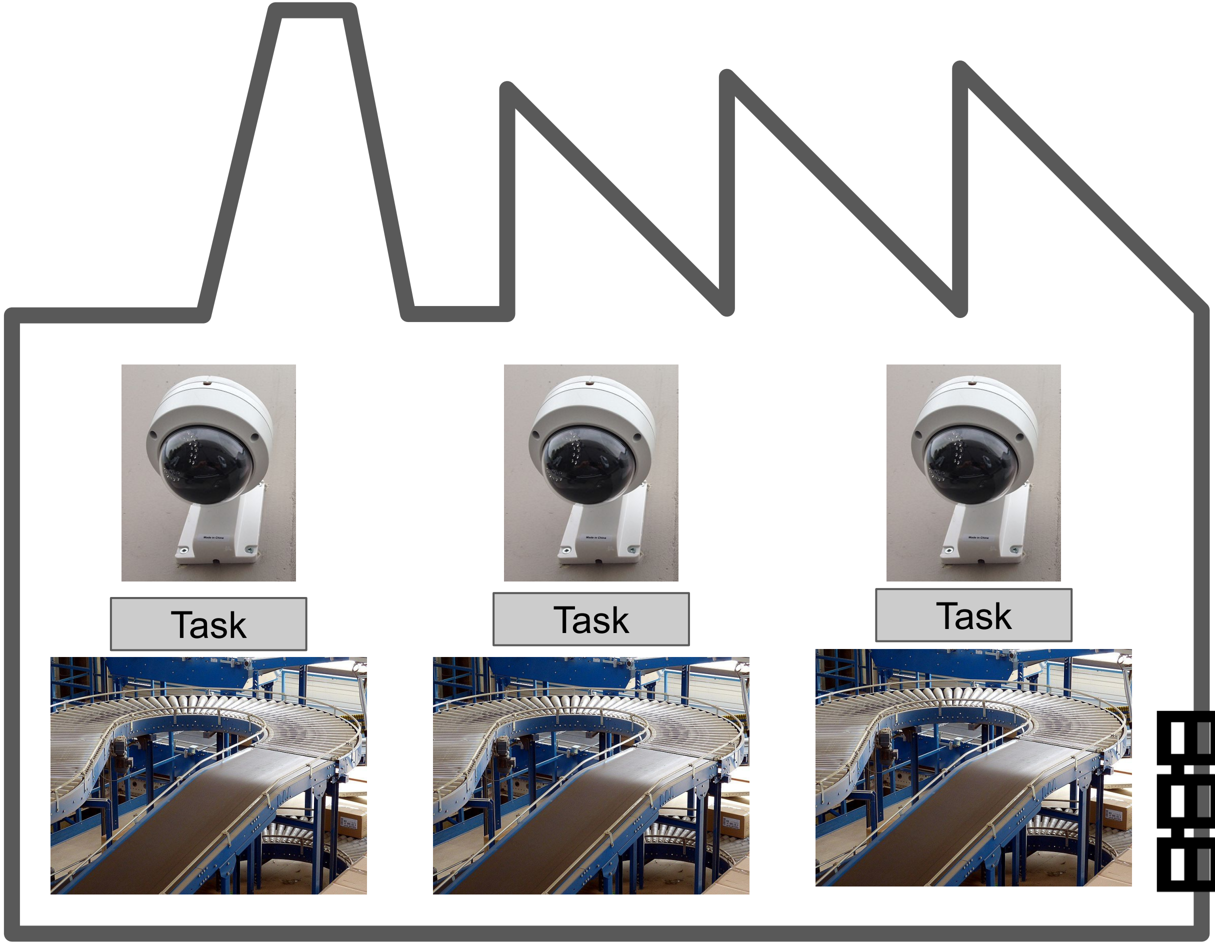
Delay constrained applications

- Ex: 10FPS processing
- <100ms for data processing on the IoT device

Current practice: offload processing to the cloud.


Cloud services
2 Data processing offloaded
to remote servers
truck
traffic light
car
person
bicycle


Task
Edge server



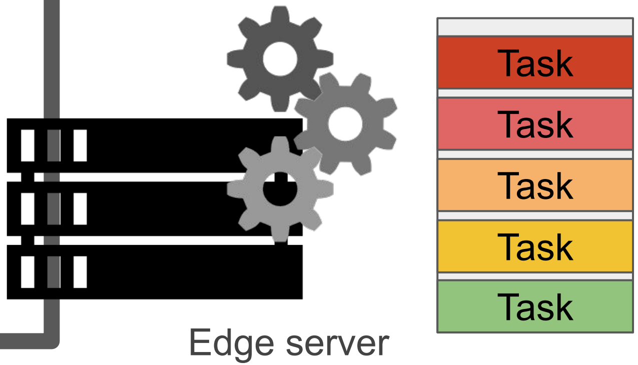
Edge compute capacity is limited.

If capacity reached => tasks dropped.

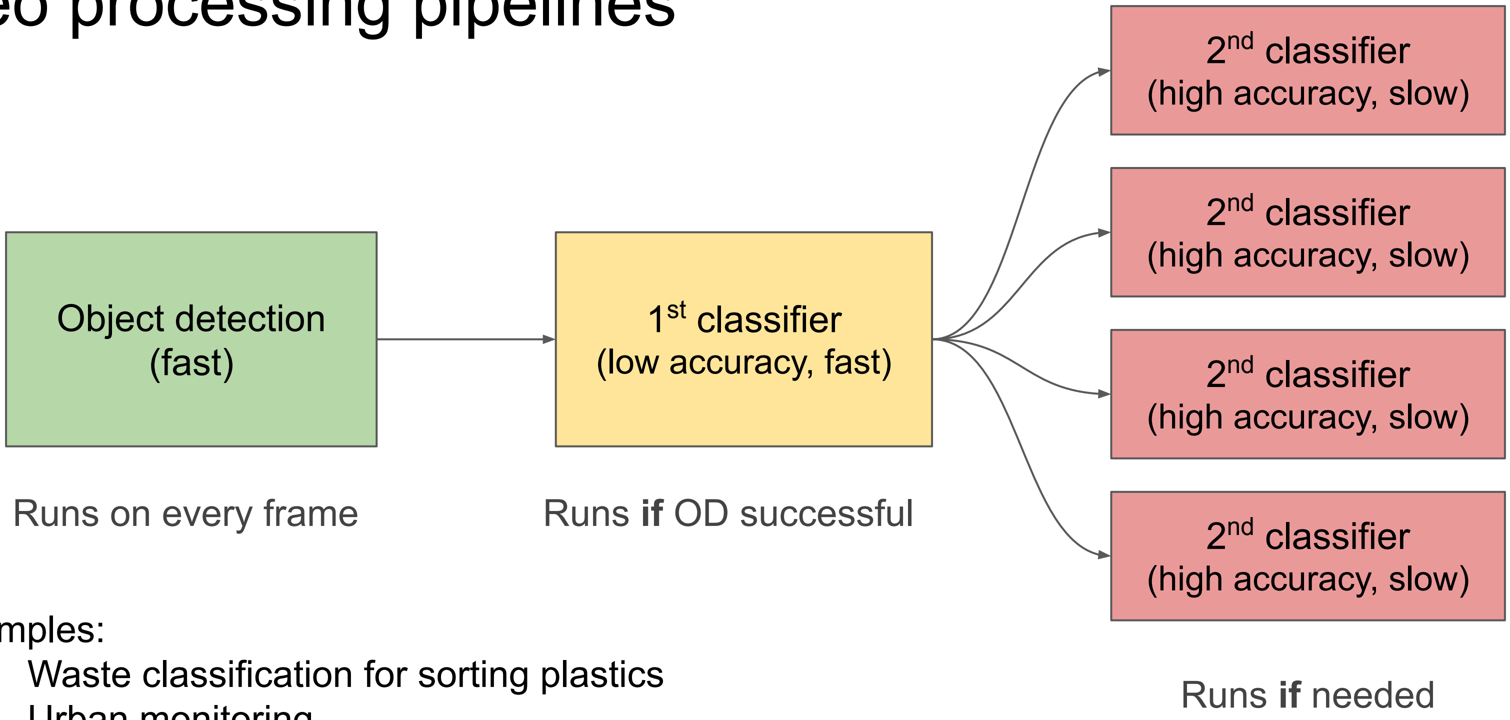
- Various bad outcomes.

Strategies for provisioning

- Mean workload
- 95%
- Over-provisioning



Video processing pipelines



Examples:

- Waste classification for sorting plastics
- Urban monitoring

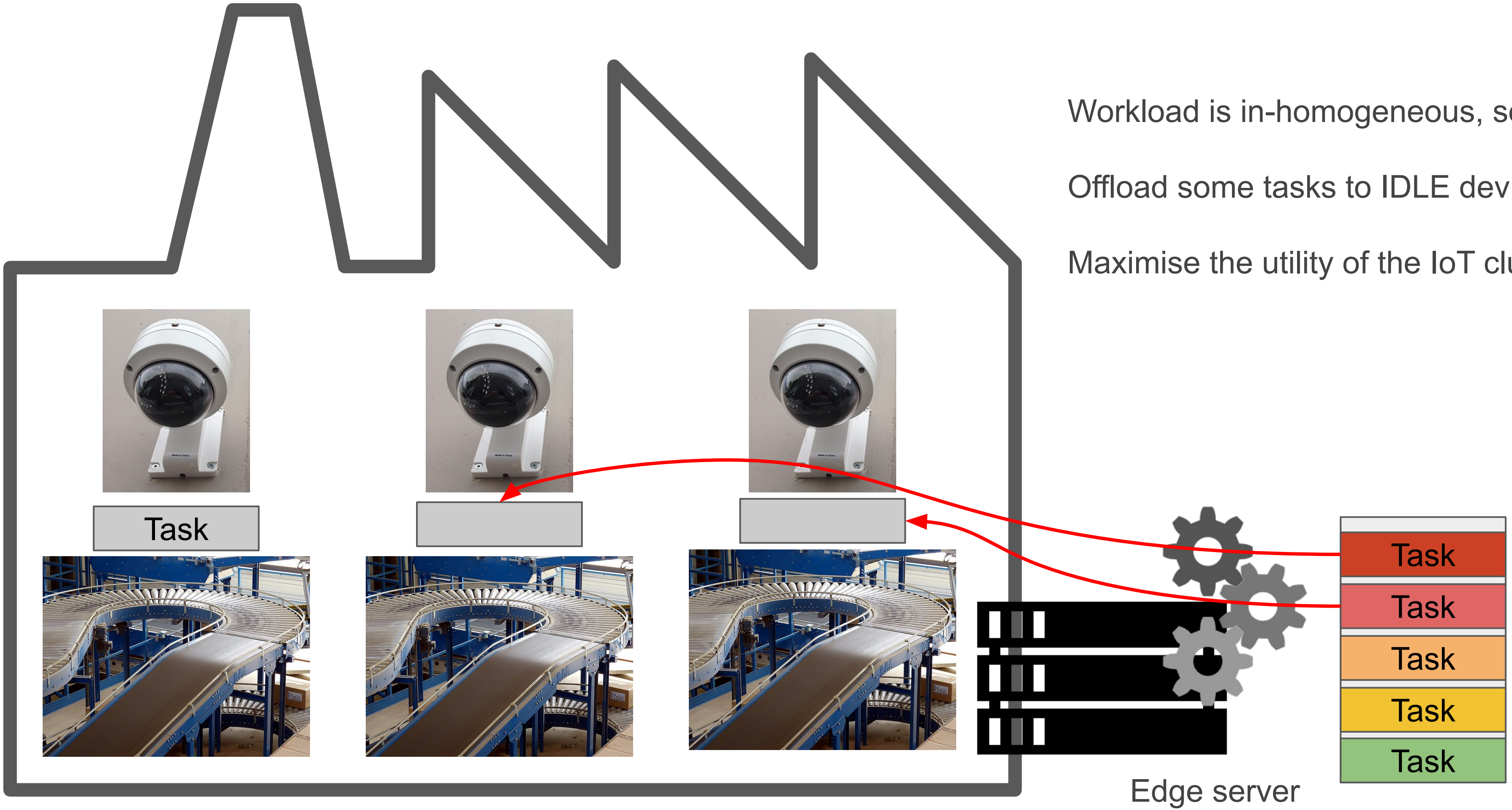
Common feature: slow, computational classifiers are only executed *sometimes*.
Per-device workload is *not homogeneous*.

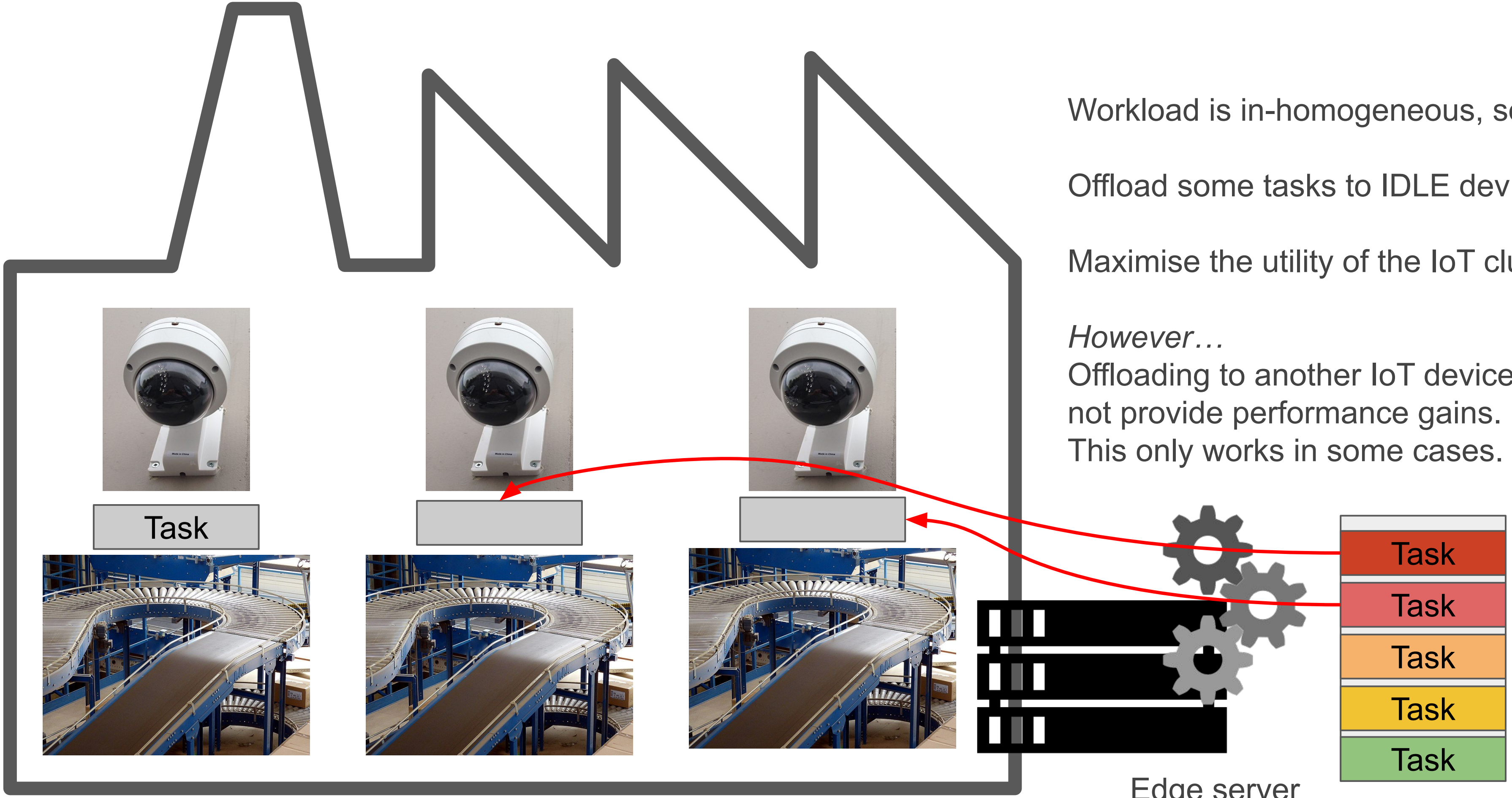
MTU Victor.Cionca@mtu.ie

Workload is in-homogeneous, so…

Offload some tasks to IDLE devices!

Maximise the utility of the IoT cluster.





Workload is in-homogeneous, so…

Offload some tasks to IDLE devices!

Maximise the utility of the IoT cluster.

However…
Offloading to another IoT device will not provide performance gains.
This only works in some cases.

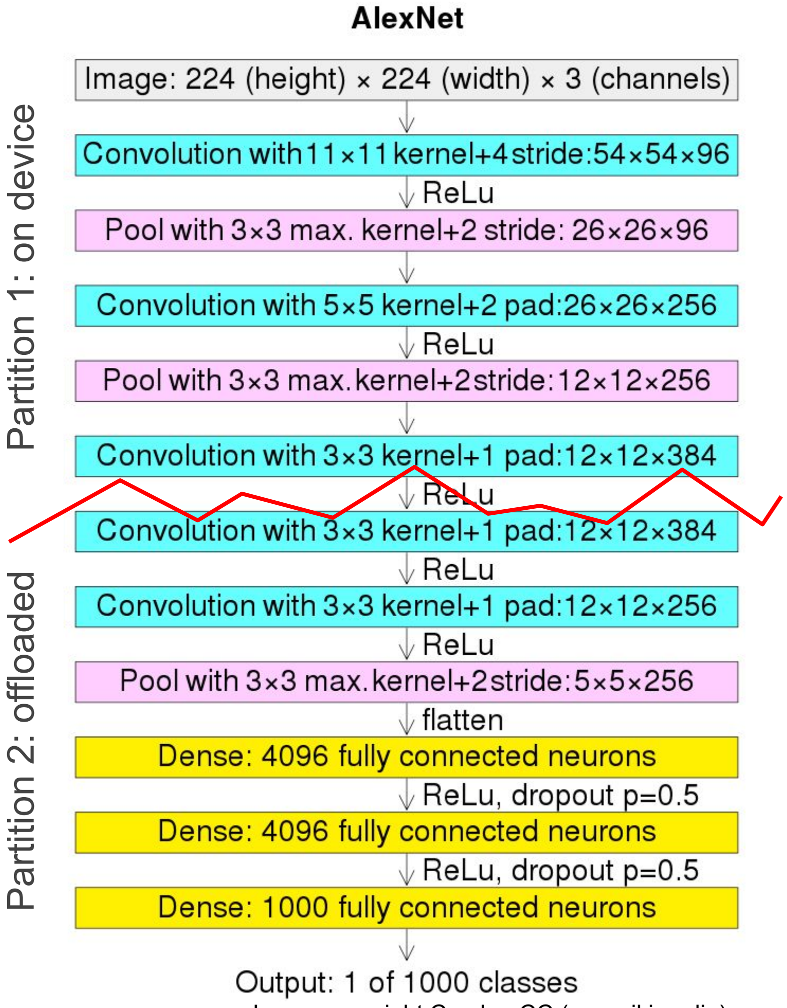
Parallelising DNNs

DNNs are predominantly sequential tasks.

Typical offloading uses *model* partitioning.

P2 must wait until P1 completes.

No gains if P2 has same computational power.



Parallelising DNNs

DNNs are predominantly sequential tasks.

Typical offloading uses *model* partitioning.

P2 must wait until P1 completes.

No gains if P2 has same computational power.

Input partitioning

- Partitions can run in parallel
- Performance gains regardless of computational power

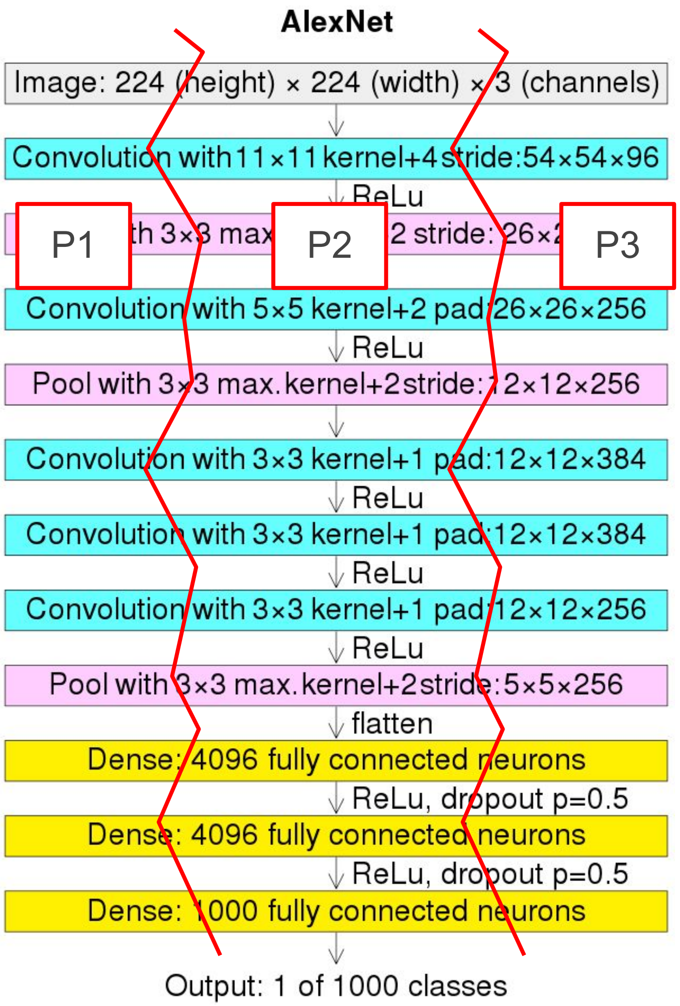
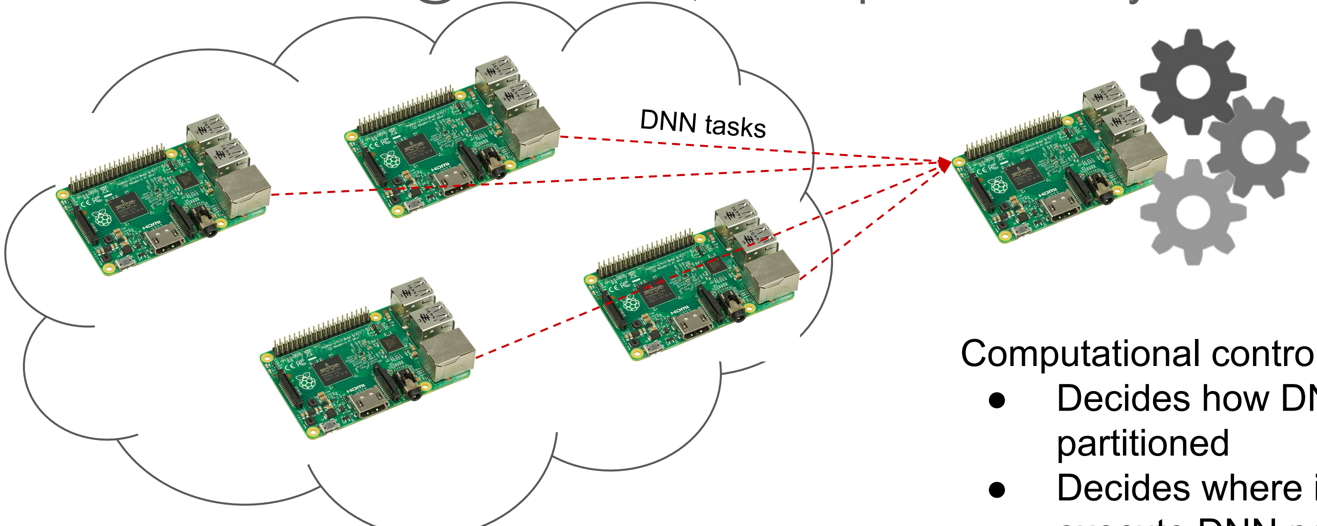


Image copyright Cmglee CC (src: wikimedia)

ADVANCECRT

Distributed inference in IoT clusters

Work with Mr. Jamie Cotter (PhD student), Dr. Ignacio Castineiras, Dr. Donna O’Shea @ MTU Cork, CS Dept. Funded by SFI ADVANCE CRT.



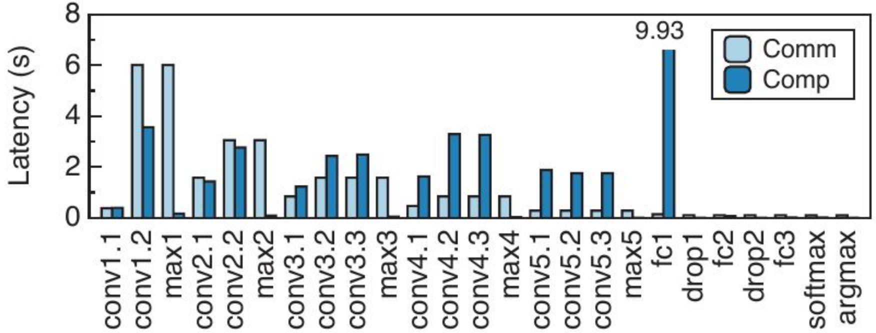
Computational controller

- Decides how DNNs should be partitioned
- Decides where in the IoT cluster to execute DNN partitions
- Computation must satisfy application delay requirements.

IoT devices: run video processing pipeline, generating DNN tasks. Can run DNN tasks of other IoT devices

Computation vs **Communication**

Image source: “Adaptive Parallel Execution of Deep Neural Networks on Heterogeneous Edge Devices”, Zhou et al, ACM SEC’19. Copyright ACM.



With computational offloading, **communication is an overhead**.

DNN inference time: ~100’s ms

Communication delays (RTT):

- WiFi <10ms
- Mobile: LTE ~50ms, 5G promised <1ms
- To nearest datacentre +10-20ms.

This is the ideal case, without communication errors.

We must improve the communication.

Faster communication

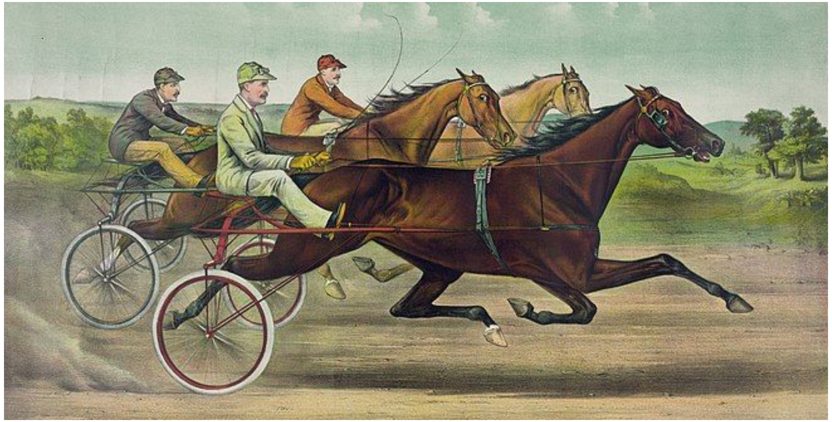
More bandwidth - example mmwave (2GHz in 60GHz band → ~35Gbps).

More transmission streams (MIMO).

Higher modulation orders (more bits per radio symbol).

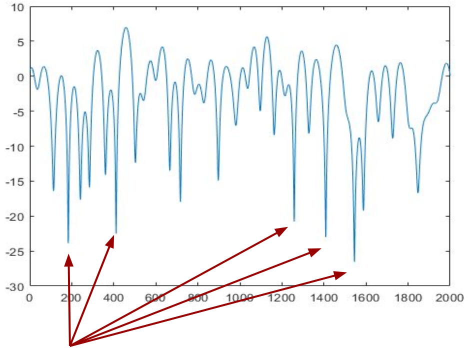
However…

Faster transmission also has higher probability of errors.



Reliability - avoiding transmission errors

Instantaneous signal power



Errors

Adding channel estimation data

Adding error checking bits

Adding guard times

Adding error correction bits

Adding complex error correction algorithms

etc.

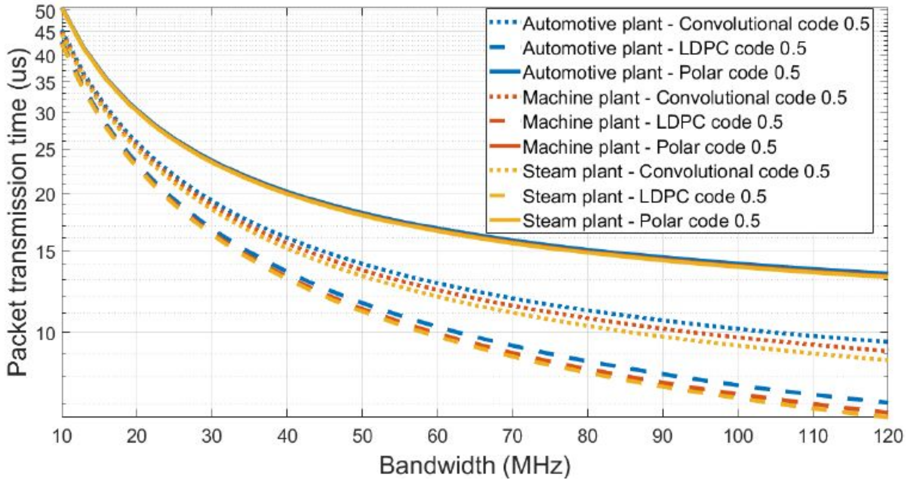
Some applications have very stringent reliability requirements.

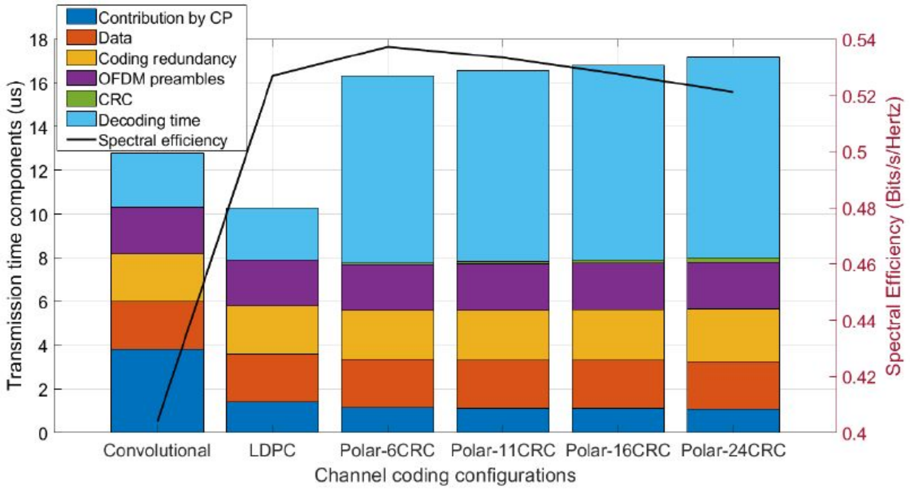
Ex industrial control loops require 10^{-9} bit error probability.

Reliability vs latency

Confirm Smart Manufacturing

Results obtained by
Mr. Yasantha Samarawickrama (PhD student)
SFI CONFIRM Centre.





Reliability and latency are most times inversely proportional.

Some alternatives: spatial diversity, cooperation, NOMA.

- Early work with PhD student Ms. Tabinda Ashraf

URLLC: Ultra Reliable, Low Latency Communication

Defined by the application.

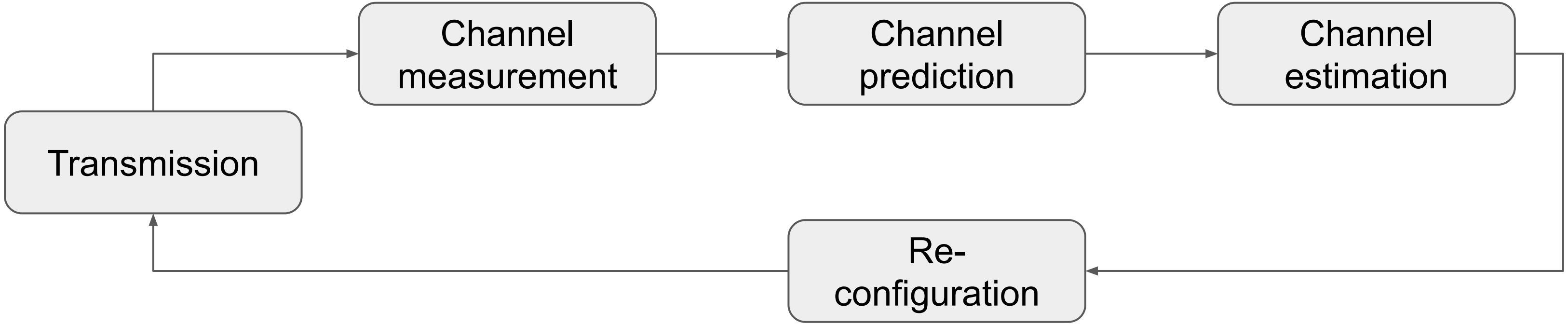
Related to *Age of Information*.

Several definitions

V1: Maximise the reliability, subject to delay constraints
V2: Minimise the latency, subject to maximum transmission error.
V3: Achieve transmission rate within given time.

We need joint optimization of reliability and latency based on channel characteristics.

Achieving URLLC



Work conducted with Ms. Tabinda Ashraf, Mr. Yasantha Samarawickrama, Dr. Álvaro de Medeiros. Funded by SFI CONFIRM Centre.

Challenge: when and how to configure the comms?

The meaning of optimal configuration

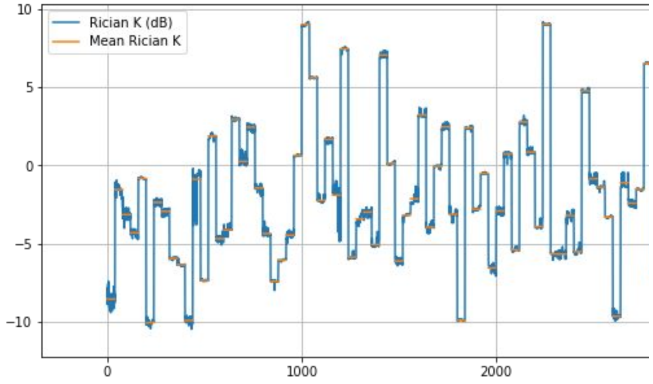
- System is configured to perform at max parameters for given channel state
- Any unexpected change in channel state leads to failures.

Basic: expected channel state.

SotA: expected distribution of channel states.

We observed that channel parameters can change drastically → different channel.

Currently investigating optimization in such scenarios.



Results obtained by Dr. Alvaro de Medeiros based on channel measurements published by NIST TN 1951.

Conclusions

- ML-based inference is still an issue in IoT
- Offloading between IoT devices possible due to non-homogeneous workloads
- Leads to more efficient resource utilization

Still challenges

- Dealing with delay constraints
- Dealing with wireless communication errors.

Thank you for your time!