7th UCAAT User Conference on Advanced Automated Testing

ETSI

DEEP LEARNING IN THE MOBILE COMMUNICATIONS TESTING DOMAIN

Presented by Carlos Cárdenas (DEKRA)

DEKRA

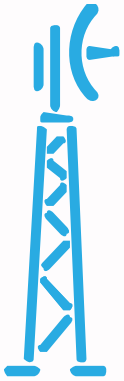
Motivation Facts

- **Drive Test** is a methodology to asses network or user equipment performance out of measurements taken by driving a vehicle or walking along a predefined route.

Drive testing measurements campaigns are time consuming and costly

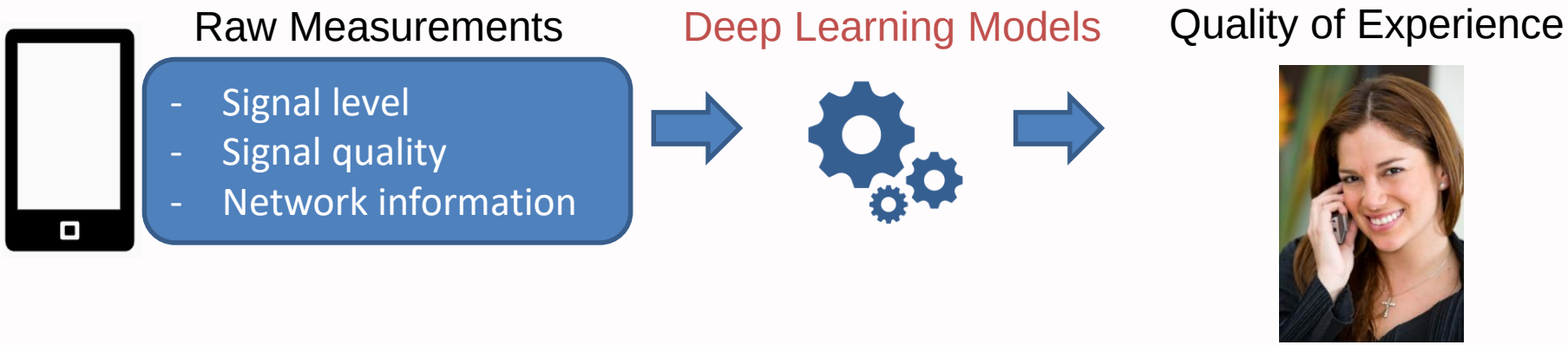
Motivation Facts

- **Minimization of Drive Test** (MDT) is a methodology promoted by the 3GPP whereby the network asks any UE around the interested area to take measurements and report them later.

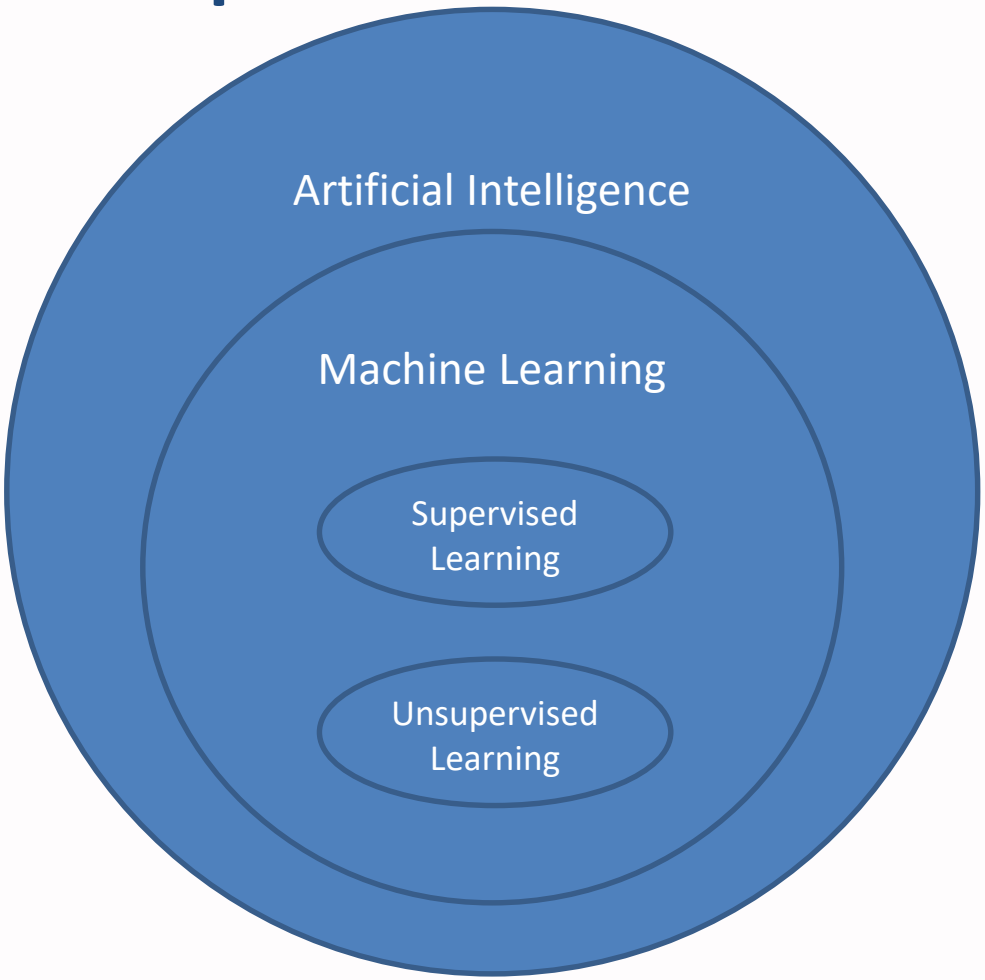


Motivation Facts

- In order to **enrich the output of MDT processes** DEKRA has developed deep learning techniques to infer **Quality of Experience (QoE) metrics** from the raw measurements collected by the UE.



Scope of this Work



Machine Leaning: A type of Artificial Intelligence (AI) that provides machine with the ability to learn without explicit programming.

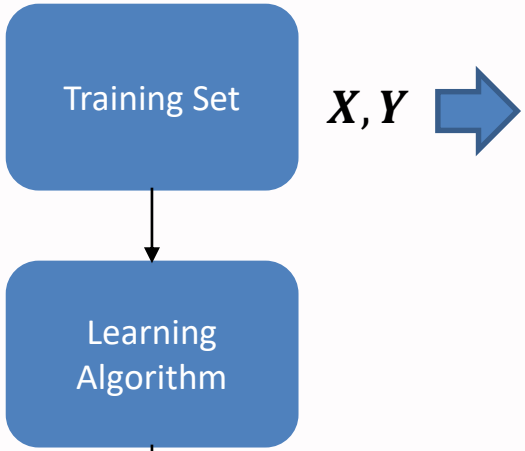
In plain words …. ML is learning by examples (dataset)

Supervised Learning: Labelled dataset: Regression, Classification

Unsupervised Learning: Unlabelled dataset: Clustering, Anomaly Detection, Feature Extraction

Supervised Learning (Basics)

Workflow



X, Y

Input Features X | Output Label Y

Size (m^2)	# bedrooms	backyard	Price
100	2	no	Low
75	3	no	Medium
250	5	yes	High
...	...	...	....

Labelled Training Set

Inference Function (production)

x → h → $\hat{y}$ (Predicted y)

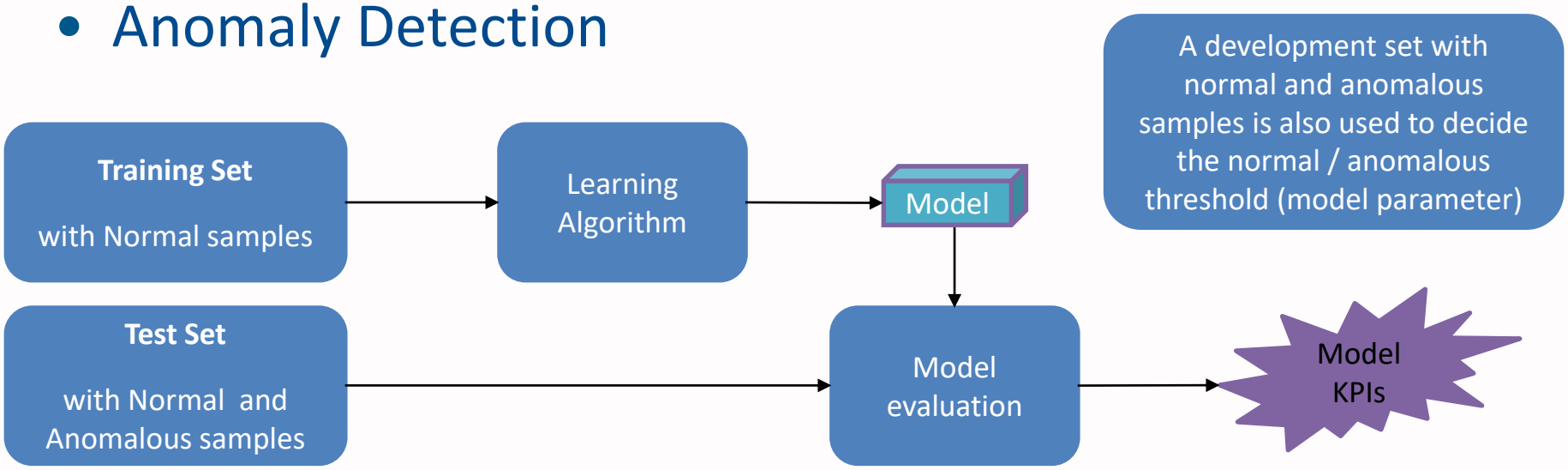
Size (m2)	# bedrooms	backyard
125	1	no

Estimated Price
Low

At least, you need two labelled dataset: training and test sets.

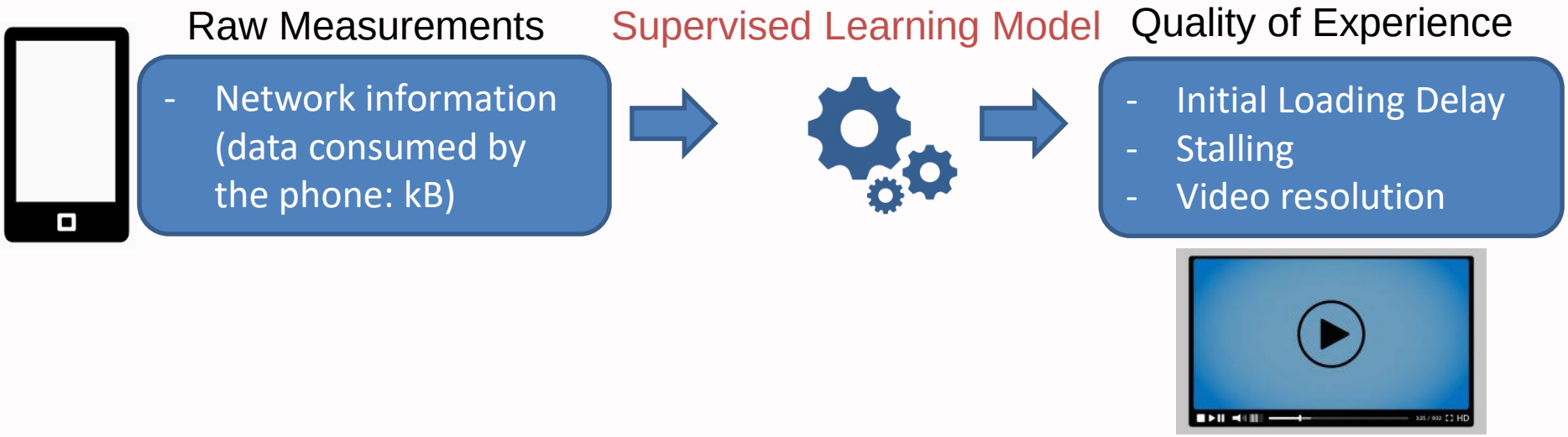
Unsupervised Learning (Basics)

- Identification of structure in the data, rare items, events or observations which raise suspicions by differing significantly from the majority of the data
- Anomaly Detection



Use Case 1: Video QoE Prediction

- **Objective**: Assessing the quality of experience of video applications using input data from a single measurement point at the smart phone



Use Case 1: Video QoE Prediction

- **Problem Formulation**
  - **Input Features X:** The feature vector $x^{(i)}$ is formed by the data amount accumulated (kilobytes) along the video session, measured every second.

Rx[1]	Rx[2]	…	Rx[n_{max}-1]	Rx[n_{max}]
10	254	…	48111	54987
15	214	…	35211	35211

n_{max} is the duration of the longest video session from the training set

One video session corresponds to one sample in the training set

Training set size: Number of test sessions executed to obtain the labelled samples

As deep learning models require rectangular matrices for training, we have used a padding technique in order to ensure that all the examples in the training set have the same number of features n_{max} regardless of the actual duration of the video session

Use Case 1: Video QoE Prediction

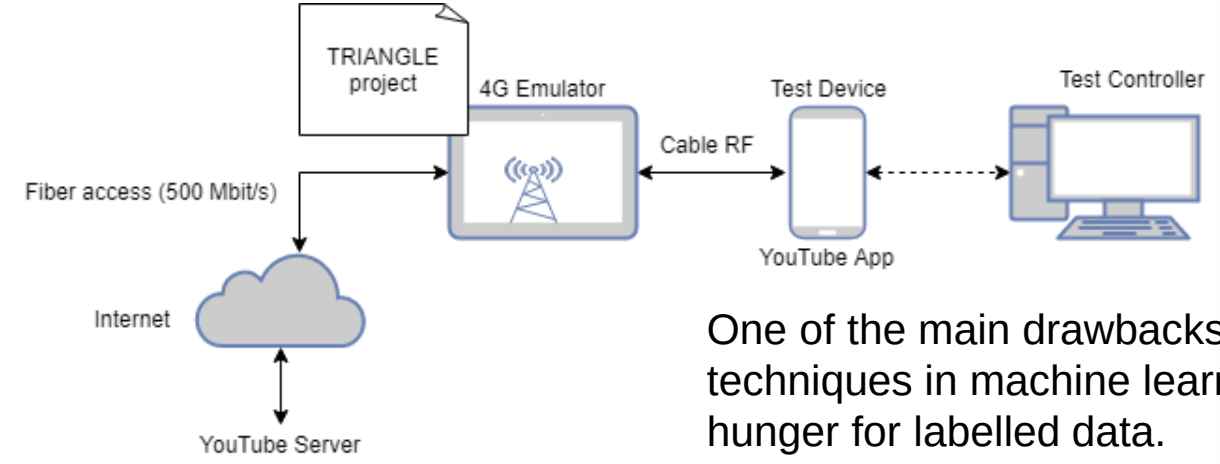
- **Problem Formulation**
  - **Output Labels Y:**
    - **Initial Loading Delay (ILD):** The time in seconds between the initiation of the video playback by the user and the actual start of the playback
    - **Total Re-buffering Time (TRB)**: The buffering events at the client side.
    - **Video resolution**: The number of horizontal lines a video is displayed from top to bottom.

Instead of predicting the exact value of the performance metric, we have discretized the labels into binary classes, good and poor, corresponding respectively to positive and negative classes in the usual deep learning nomenclature.

Negative Classes Thresholds
- ILD ≥ 4
- TRB > 0
- Video resolution < 1080

Use Case 1: Video QoE Prediction

- **Training dataset generation**



One of the main drawbacks of the supervised techniques in machine learning comes from their hunger for labelled data.

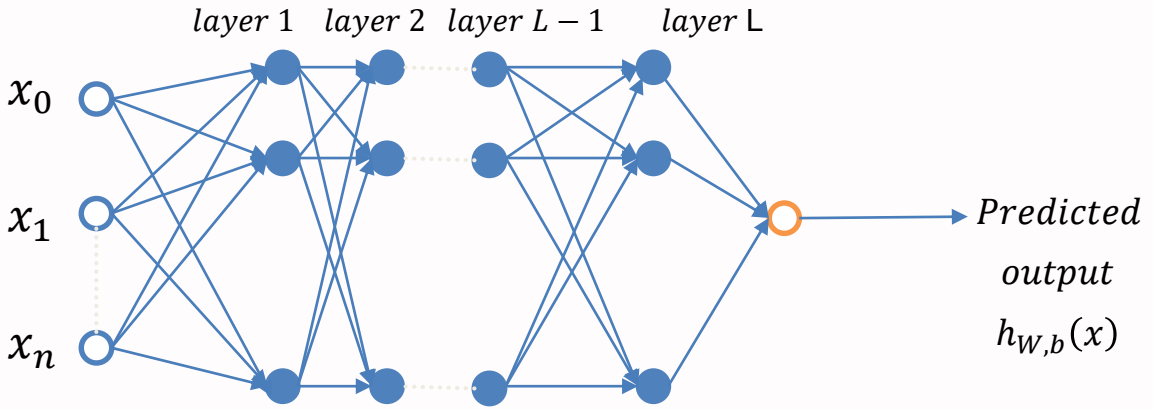
Relevant information about the Training
80 video clips
Duration of selected video clips: 180 s
Longest video session: **378** s (n_{max})
video of sessions (training set size): **1040**

In this work, a test tool named TACS4 has been used to **automatically generate the set of labelled data**. TACS4 employs Appium framework to recognize the user interface artifacts which appear on the smartphone screen such as spinning wheels, buttons, etc.

Use Case 1: Video QoE Prediction

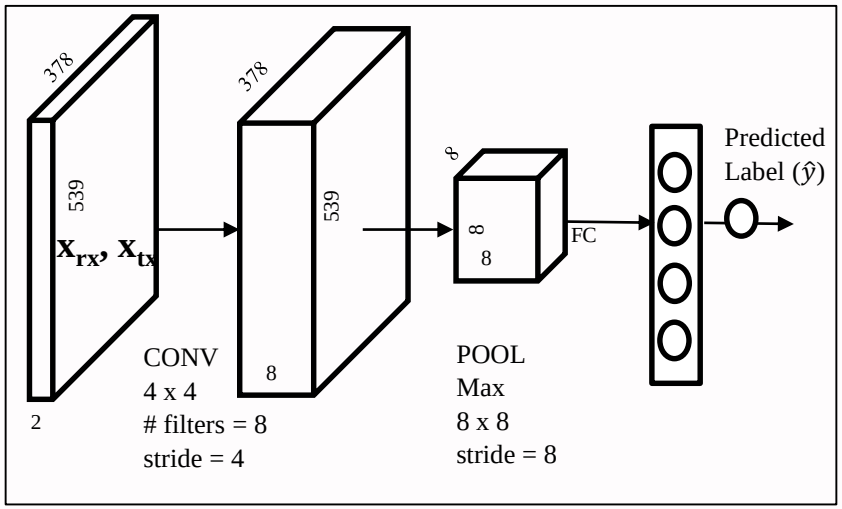
- **Evaluated Neural Network Architectures**
  - **Single task Fully Connected Neural Network**
  - **Multi-task Fully Connected Neural Network**
  - **Convolutional Neural Network**

Neural Network (Basics)
A neural network consists of a layered architecture of activation nodes. The activating nodes among consecutive layers are connected through links, which are linear functions defined by their weights W and bias b. An activation node or neuron is a non-linear function (e.g. $f(z) = (1+e^{-z})^{-1}$) which is applied to the linear combination of its inputs.



Use Case 1: Video QoE Prediction

- **Convolutional Networks**

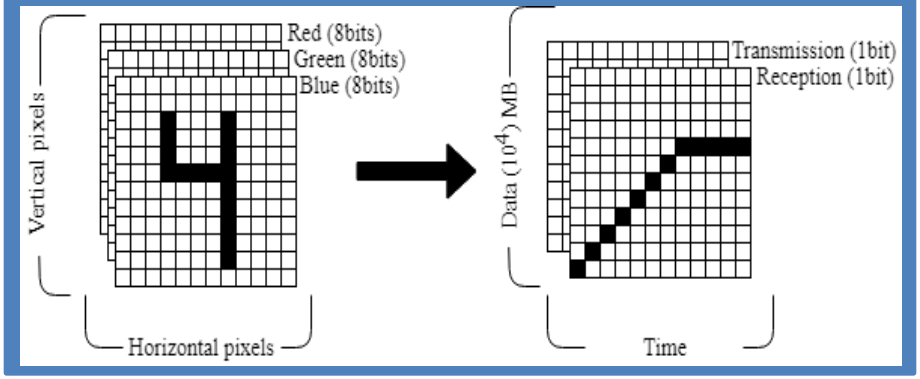


This approach has been inspired by computer vision use case

Convolutional neural network (CNN) is a specialized kind of neural network often used in computer vision. CNNs employ a mathematical convolution instead of general matrix multiplication in at least one of their layers.

CNNs are able to process simultaneously several channels, that is, n_c input matrices with each element containing the value of the intensity for one pixel.

In our problem, we use two channels: Transmitted and received data.



Use Case 1: Video QoE Prediction

- **Results**

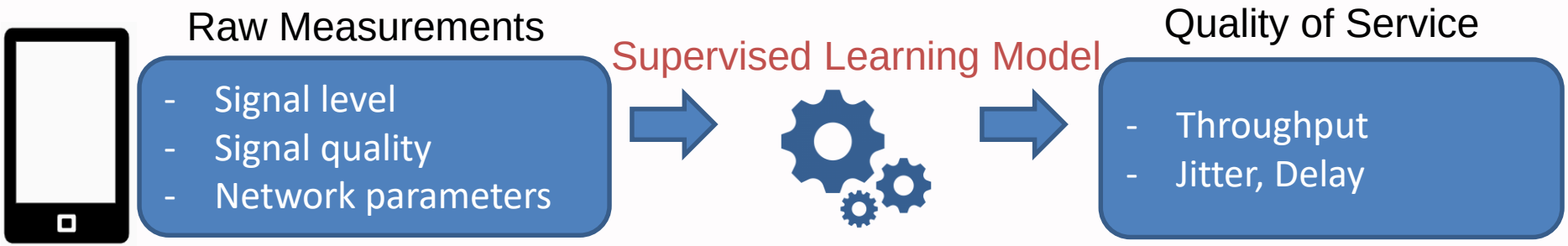
Task	Accuracy (%)
Resolution (Mode)	89.9 %
Resolution (Average)	90.1 %
Initial Delay	92.3 %
Total Rebuffering Time	88.8 %

Three deep neural network architectures were implemented, providing an overall **performance of 90%** on the test set.

This is a promising result taking into account that we only used 80 video clips to train the classifiers, and that the generation of new labelled datasets is feasible at low cost thanks to the fully automated test setup used.

Use Case 2: QoS Prediction (stationary)

- **Objective**: Assess the QoS of the radio network (in stationary conditions) using network parameters from non-intrusive passive monitoring.



Use Case 2: QoS Prediction (stationary)

- **Problem Formulation**
  - **Input Features X:** The feature vector $x^{(i)}$ is formed by aggregated statistics of the network parameters extracted from the smart phone traces during a test session.

$x^{(i)}$
# of handovers
Min/Avg/Max LTE RSRP (dBm).
Min/Avg/Max LTE Strongest Neighbour RSRP (dBm).
Min/Avg/Max LTE Timing Advanced
….

We needed to do some **data preparation** before training the model. For example, handling missing data when there are no neighbour cells.

Number of features: **18**

Use Case 2: QoS Prediction (stationary)

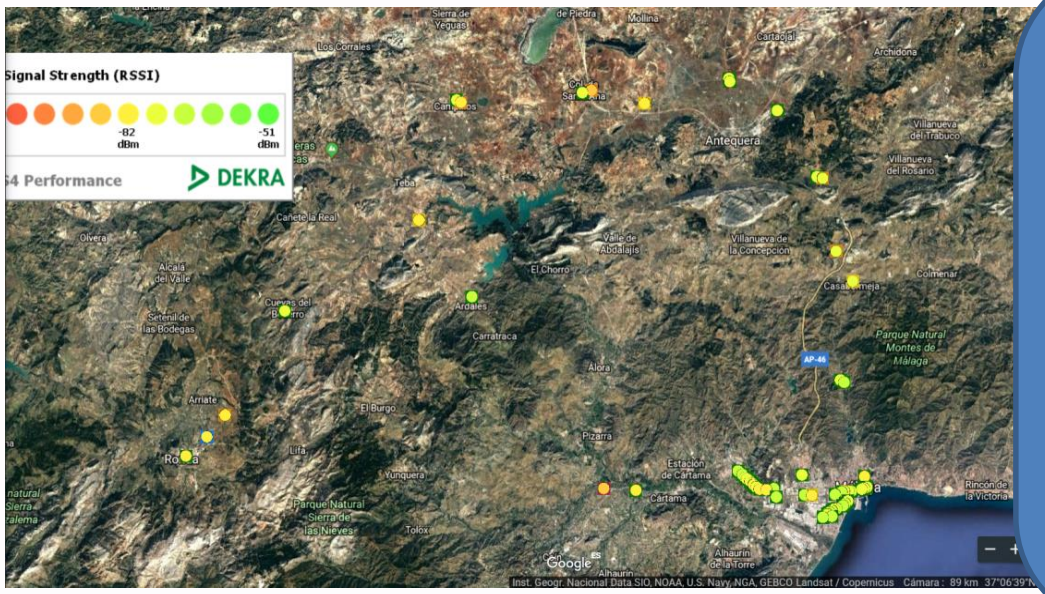
- **Problem Formulation**
  - **Output Labels Y:**
    - **Throughput Downlink (Mbit/s)**: Poor, Fair, Good
    - **Throughput Uplink (Mbit/s)**: Poor, Fair, Good
    - **Delay/Jitter**: Poor, Fair, Good

Instead of predicting the exact value of the performance metric, we have discretized the labels into multiple classes, poor, fair, and good.

In order to get **a balanced set** we have adjusted the threshold to percentile 33th and 66th. This way we have 33% samples of each class.

Use Case 2: QoS Prediction (stationary)

- **Training dataset generation**
  - **Manual stationary field testing**



Manual (costly) generation of samples:

Repeat for each test location

1. Drive to specific position
2. Run 180 seconds test session
3. Aggregate network information and generate input features for this sample.

Number of test sessions (training set size): 163

Use Case 2: QoS Prediction (stationary)

- **Results**

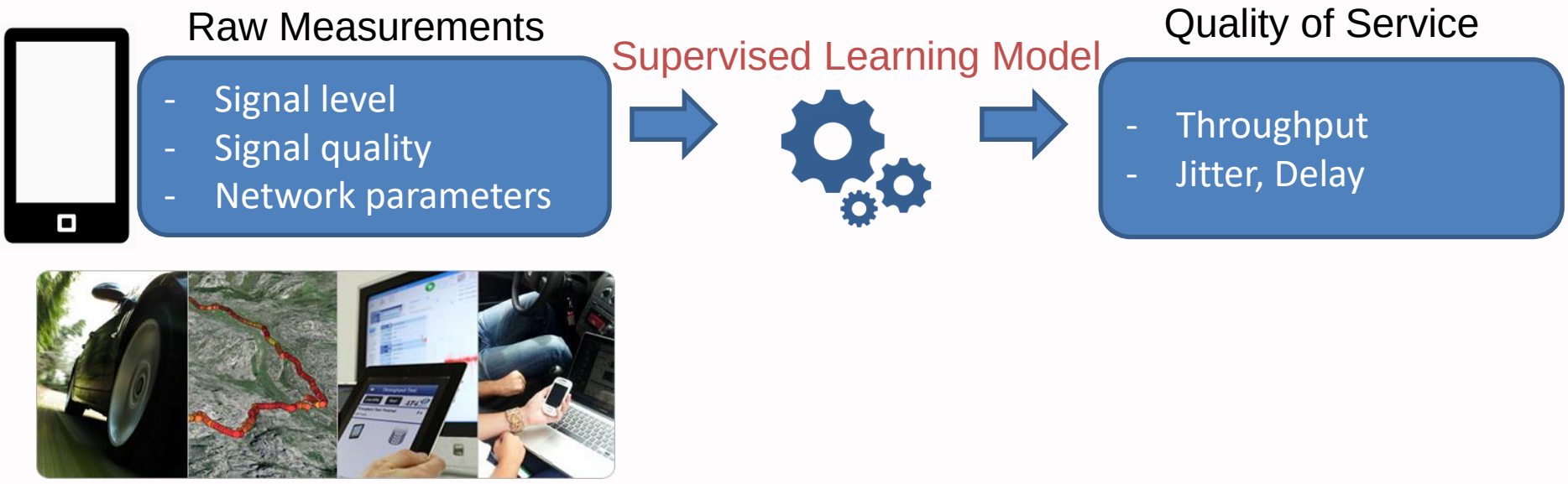
Task	Accuracy (%)
TCP DL Throughput	90.9 %
TCP UL Throughput	97.7 %
UDP Delay / Jitter	86.2 %

Even though the results are good, the scope of this approach is limited to stationary test scenarios. Also, the generation of new training samples is costly.

We have used Model Regularization techniques to avoid that the model overfits the training data and gives low performance on the production samples

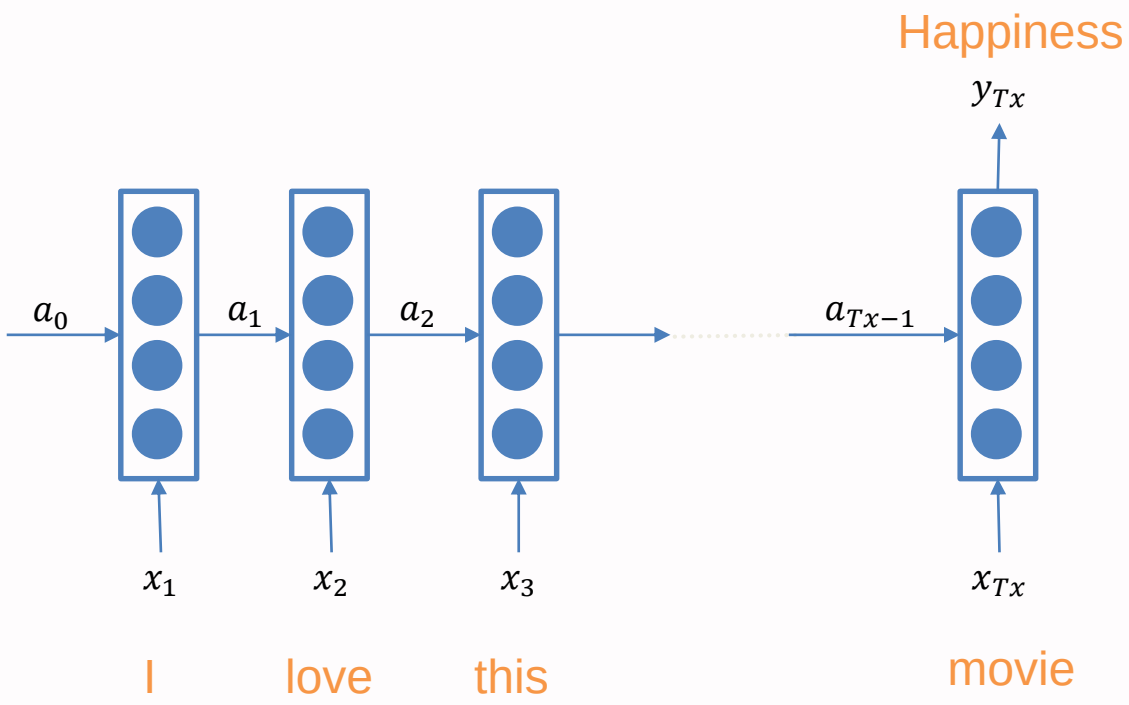
Use Case 3: QoS Prediction (mobility)

- **Objective**: Assess the QoS of the radio network (in mobility conditions) using network parameters from non-intrusive passive monitoring.



Use Case 3: QoS Prediction (mobility)

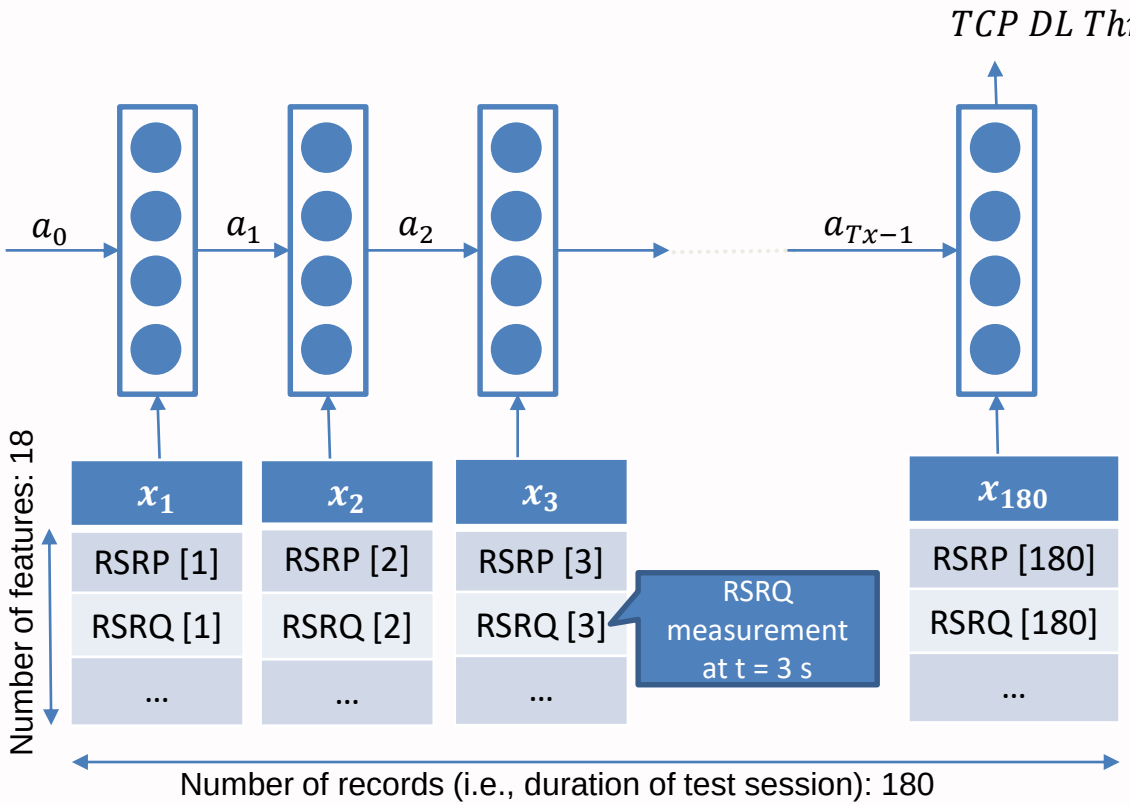
- **Recurrent Neuronal Networks**



- **RNNs** for addressing sequential data problems
- The training data consists of sequence $(\boldsymbol{x}, y)$ pairs, where these sequences exhibit significant correlation.
- Typical Applications: Sentiment analysis, Voice Recognition, Machine Translation

Use Case 3: QoS Prediction (mobility)

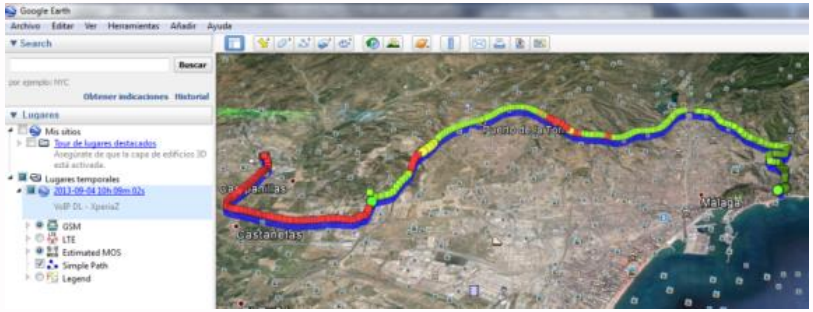
- **Problem Formulation**



- We have used many-to-one RNN to make our approach valid for **mobility test scenarios**.
- The Input Features "X" are no longer an statistical aggregation but the measurement records taken every second. This way **the model has the potential to learn the network dynamics**.

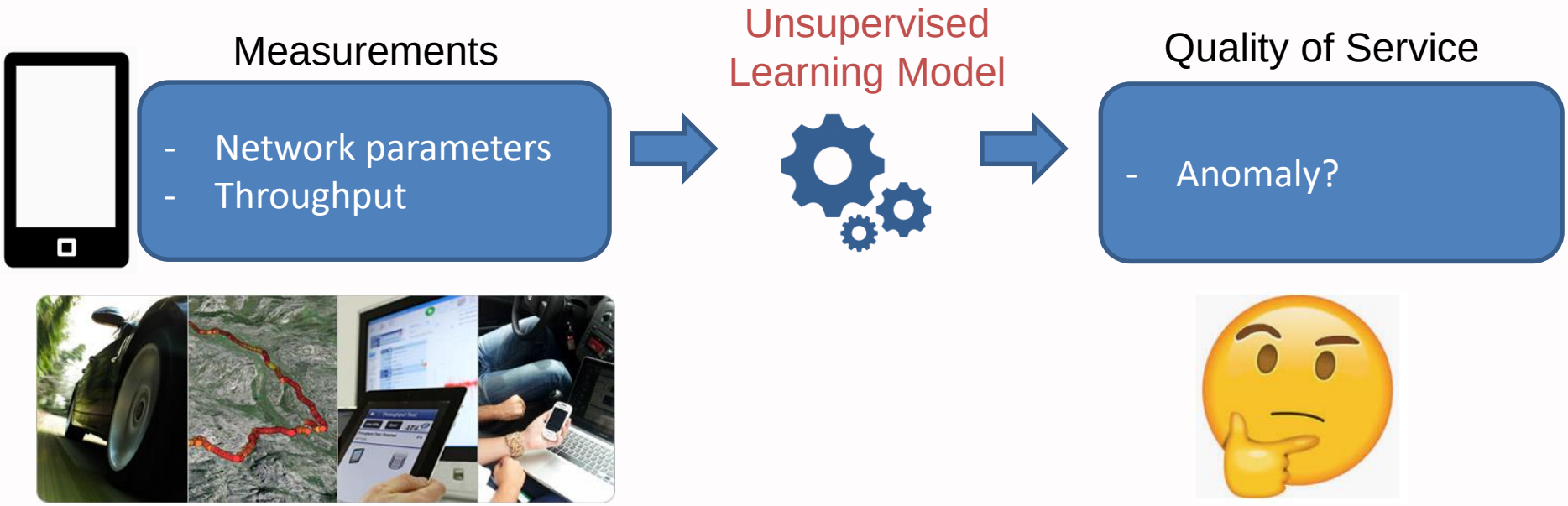
Use Case 3: QoS Prediction (mobility)

- **Results** 
  - This model is still under development
  - Preliminary results are promising: 75% accuracy
  - Main advantages:
    - It captures network dynamics
    - Training set generation is less costly than stationary case because you just drive along the route while taking measurements



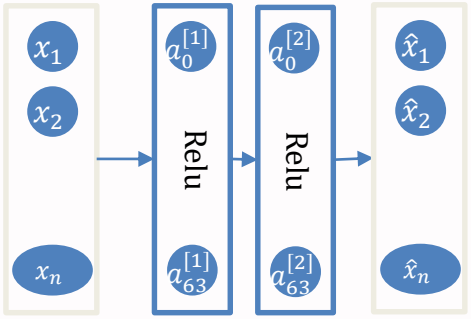
Use Case 4: Anomaly QoS Detection

- **Objective**: The goal of the system is to detect marginal throughput test results given specific network conditions.



Use Case 4: Anomaly QoS Detection

- **Anomaly Detection based on auto-encoder**



Original input Reconstructed input

Reference: https://arxiv.org/abs/1811.05269

"The key idea is to train a set of auto-encoders to learn the normal (healthy) behaviour of the system and, after training, use them to identify abnormal conditions."

"If we feed a trained auto-encoder with data not seen during training, it cannot reconstruct the input with good fidelity"

Training:
Using normal examples, learn the model W, b parameters to minimize:

$$J(W,b) = Loss(x,\hat{x})$$

Detection:
1. Given a new example x_{test}, compute Reconstruction Error $= Loss(x_{test},\hat{x}_{test})$
2. Anomaly if $RE > \varepsilon$ (e.g., 0.02)

Use Case 4: Anomaly QoS Detection

- **Problem Formulation**

Input Features

$x^{(i)}$
Avg. Throughput (Mbit/s)
Avg. RSRP (dBm)
Avg. RSSI (dBm)
Avg. RSRQ (dB)

A sample in the training set corresponds to a 180 seconds throughput test session

Training Set Generation

Training Set

with Normal samples

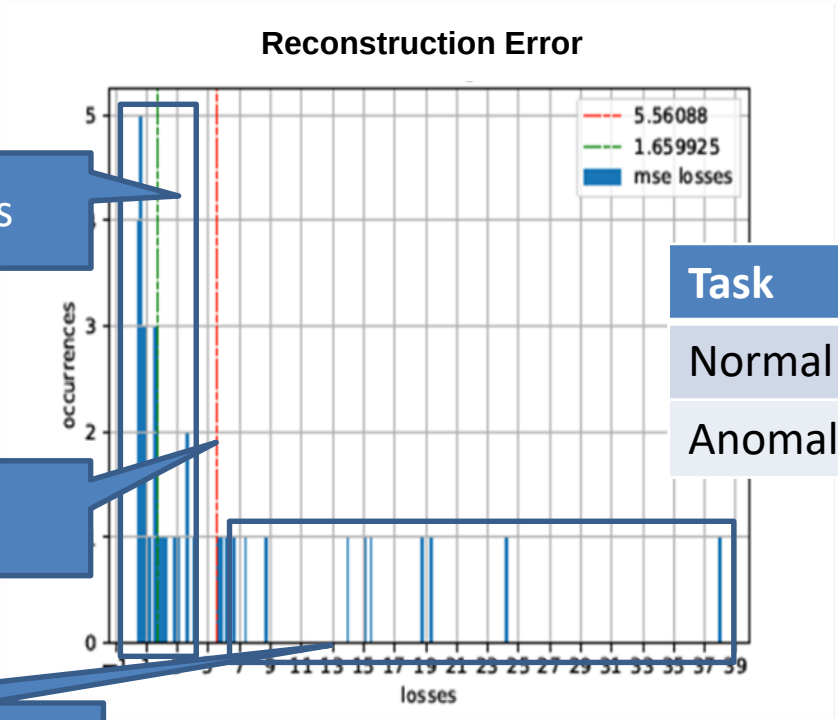
Test Set

with Normal and Anomalous samples

Number of test sessions (training set size): 163

Use Case 4: Anomaly QoS Detection

- **Results**



Detected as Normal samples

Decision Threshold

Detected as Anomalous samples

Task	Accuracy (%)
Normal	100 %
Anomalous	97.7 %

Conclusions

- One of the main challenges to apply Deep Learning in drive testing is **the correct definition of the problem in terms of machine learning**: mapping of input features to output values (or classes). In or our cases, from network radio parameters to QoE metrics.
- Measuring real apps QoE in production networks imply testing the end to end data path in which the **public Internet** introduces uncertainty. This **means noise in the prediction models** that need to be addressed.
- Use **multi-layer perceptron (MLP)** architecture (fully connected) to deal with problems where the **input features are not correlated** (e.g., UE radio measurements).
- Use **convolutional (CONV) network** architecture to deal with problems where the **input features show some spatial correlation**.
- Use **RNN** to deal with problems where the **input features show sequence correlation** (e.g., mobility scenarios).

Acknowledgments

- **This work has been developed inside the scope of**
  - The Technological Corporation of Andalusia (CTA) under Grant 17/956; and by the Spanish Government and FEDER under Grant TEC2016-80090-C2-1-R.)
  - The co-founded Horizon 2020 European Union project TRIANGLE

TRIANGLE Project

5G Applications and Devices Benchmarking

Thank you!