# PS-008:
# Network failure classification model using network digital twin

Junichi Kawasaki
KDDI Research/KDDI

**Acknowledgement:**

This work was conducted as part of the project entitled "Research and development for innovative AI network integrated infrastructure technologies (JPMI00316)" supported by the Ministry of Internal Affairs and Communications, Japan.

# Background

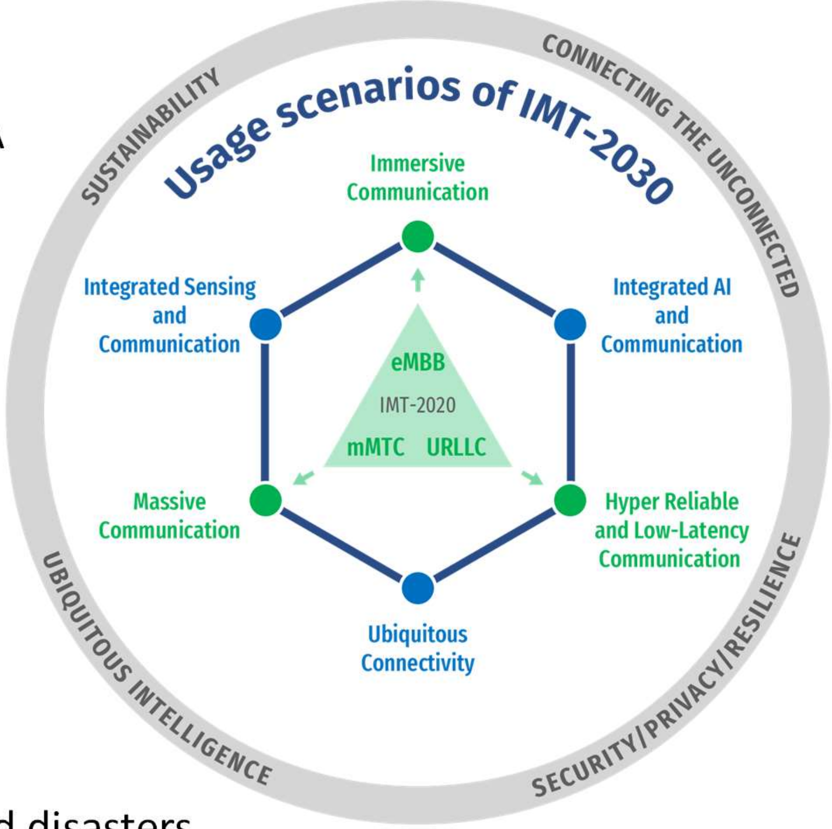
KDDI Research

## ■ Challenge in 5G and Beyond 5G (B5G)

- 5G
  - eMBB, URLLC, mMTC
  - Network Slicing for integrating these features according to SLA
  - Virtualization (VNF, CNF)

- Challenge
  - A greater number of components in networks
  - Increase in operational workloads

- IMT-2030
  - Extended features of eMBB, URLLC, mMTC
  - New aspect: Sensing, AI, Ubiquitous
  - Combination of features for respective scenario
  - Same challenge would remain in B5G
  - How to establish resilient networks against such as failures and disasters

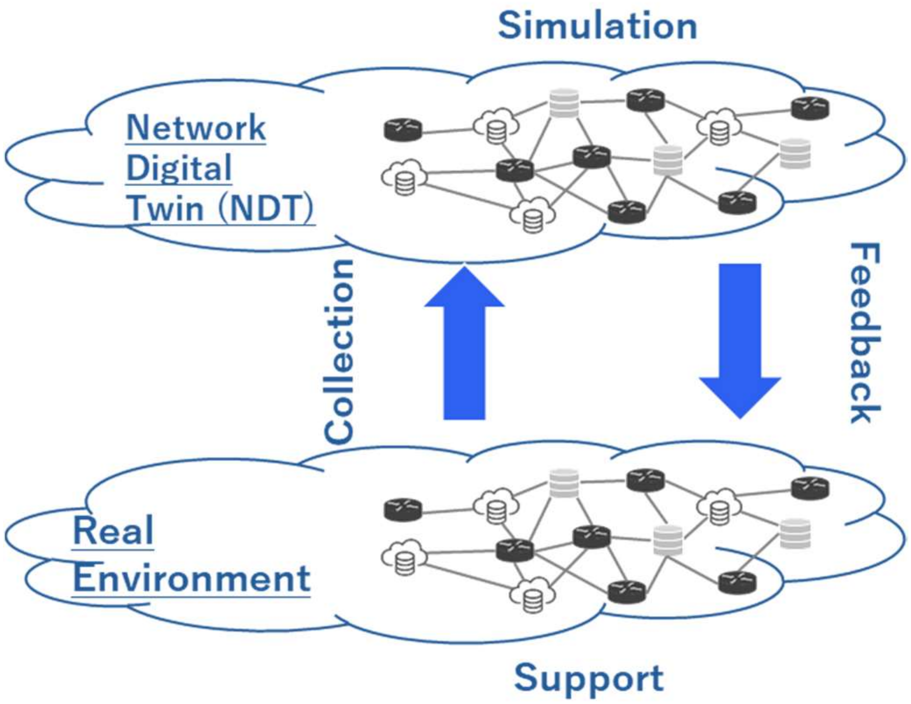

Usage scenarios of IMT-2030
SUSTAINABILITY
CONNECTING THE UNCONNECTED
UBIQUITOUS INTELLIGENCE
SECURITY/PRIVACY/RESILIENCE
Immersive Communication
Integrated Sensing and Communication
Integrated AI and Communication
eMBB
IMT-2020
mMTC
URLLC
Massive Communication
Hyper Reliable and Low-Latency Communication
Ubiquitous Connectivity

ITU-R, IMT.FRAMEWORK FOR 2030 AND BEYOND

# Background

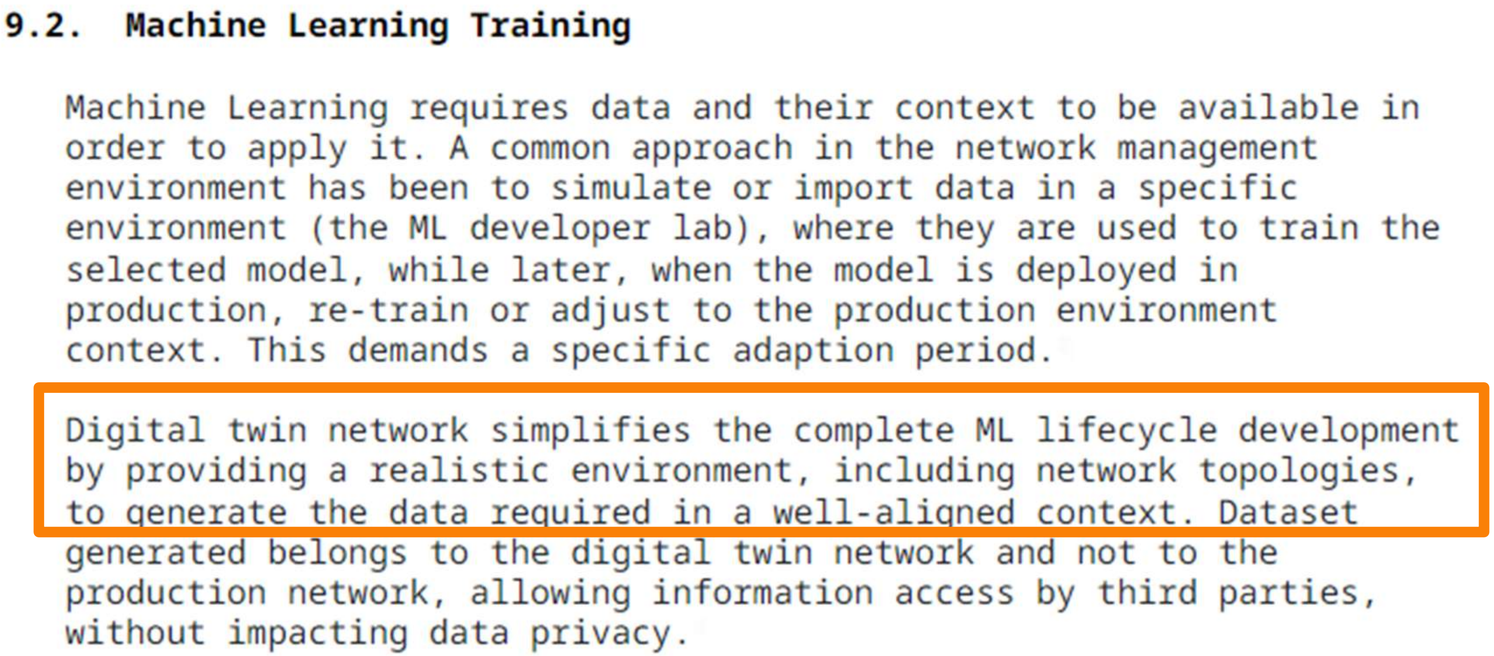
## ■ Network Digital Twin

- Network Digital Twin (NDT) is a mirror network for a real network for supporting actual operations in a real environment.
- Step in using NDT
  1. Collect network information from a real environment
  2. Perform a variety of practical simulations on NDT
  3. Feedback to support operational tasks on a real environment
- Use Case in IRFT document(*)
  - Human Training
  - Machine Learning Training
  - DevOps-Oriented Certification
  - Network Fuzzing
  - Network Inventory Management



*draft-irtf-nmrg-network-digital-twin-arch

# Use Case: Machine Learning Training

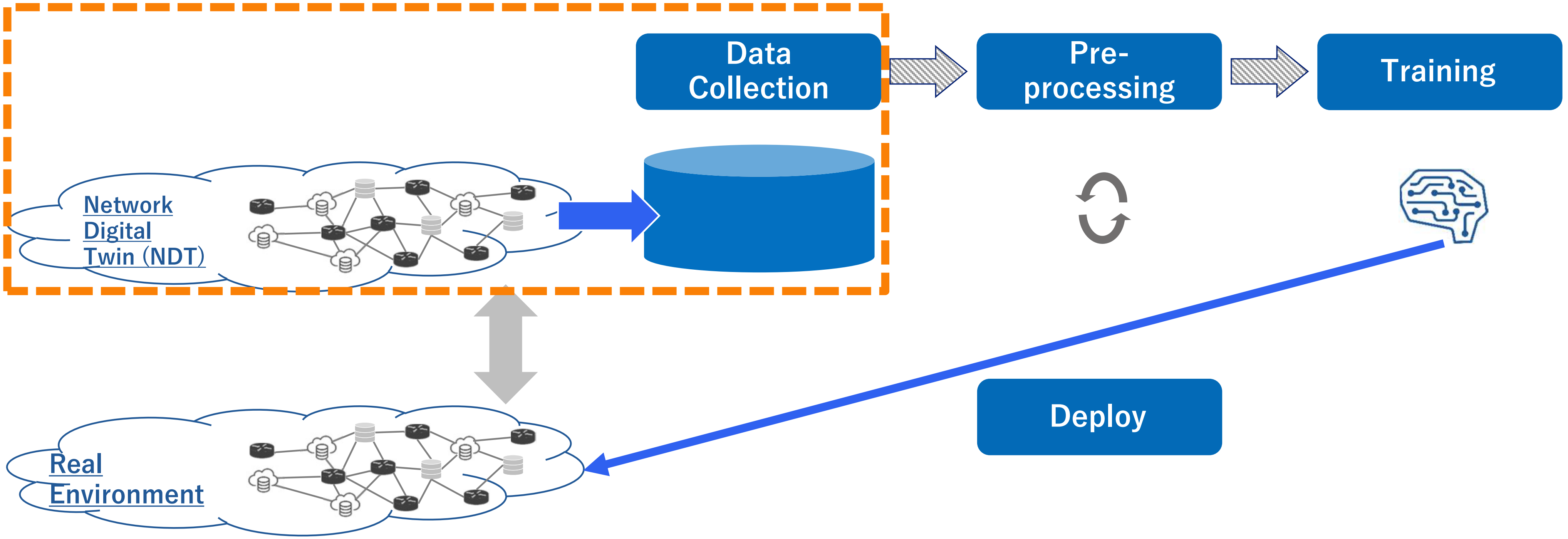


### 9.2. Machine Learning Training

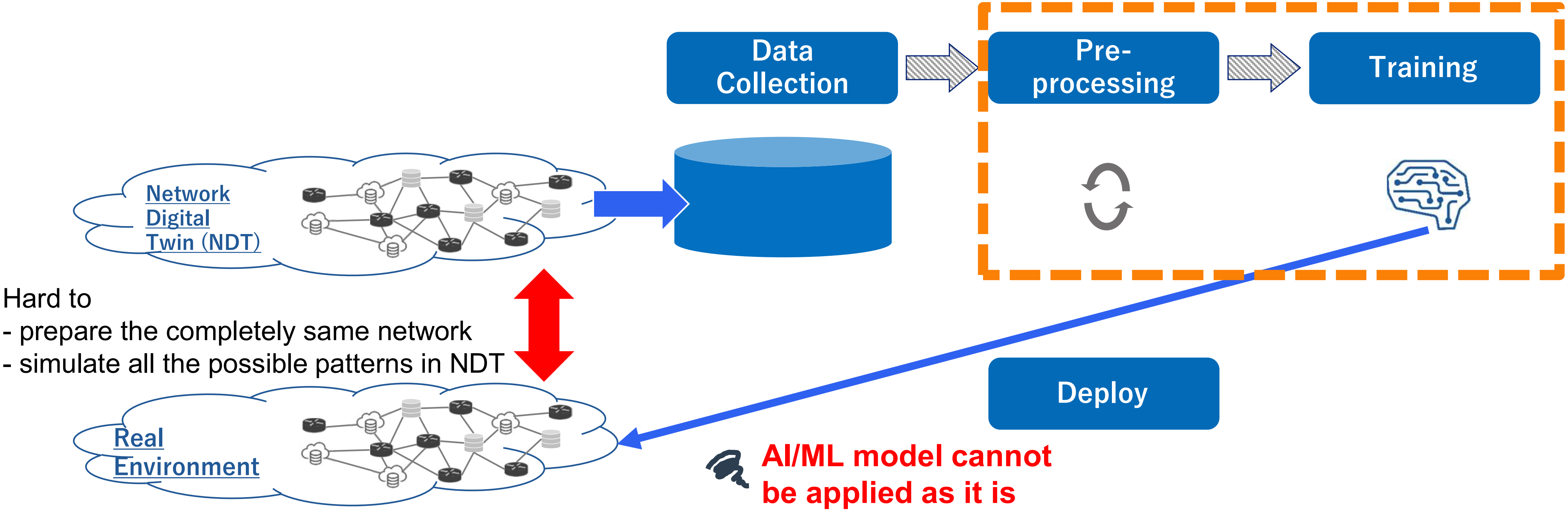
Machine Learning requires data and their context to be available in order to apply it. A common approach in the network management environment has been to simulate or import data in a specific environment (the ML developer lab), where they are used to train the selected model, while later, when the model is deployed in production, re-train or adjust to the production environment context. This demands a specific adaption period.

Digital twin network simplifies the complete ML lifecycle development by providing a realistic environment, including network topologies, to generate the data required in a well-aligned context. Dataset generated belongs to the digital twin network and not to the production network, allowing information access by third parties, without impacting data privacy.

# Challenge for ML Training on NDT



# Challenge for ML Training on NDT



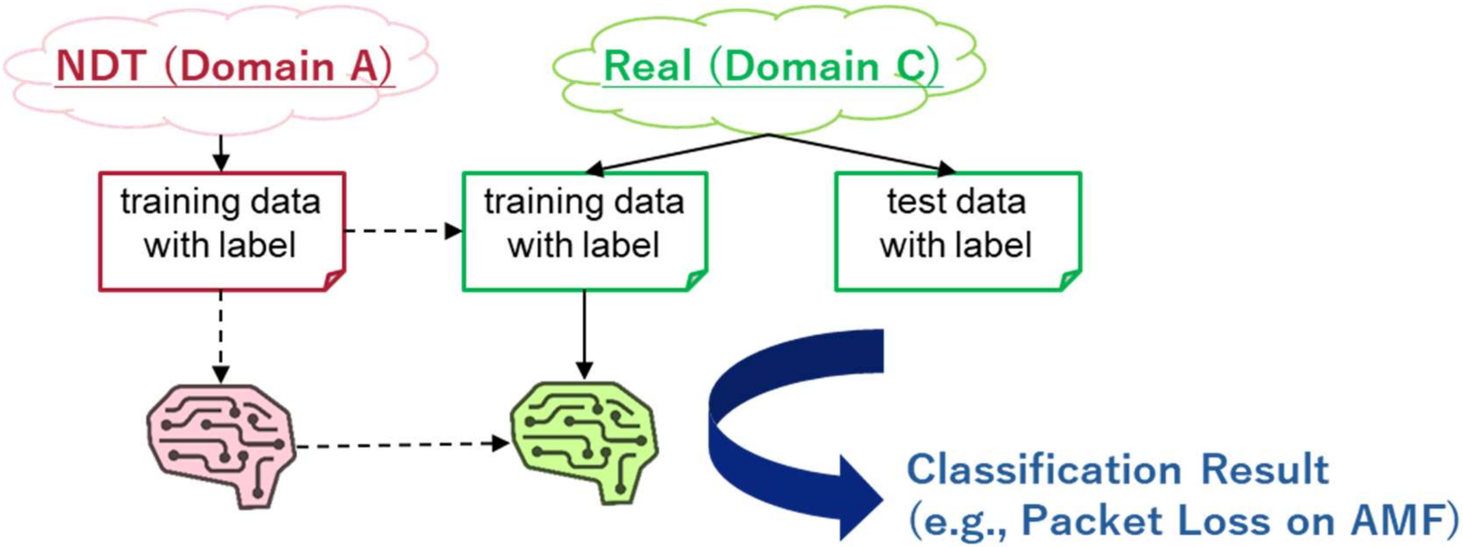
Hard to
- prepare the completely same network
- simulate all the possible patterns in NDT

AI/ML model cannot be applied as it is

**The approach to create ML models from NDT data should be somehow refined to obtain good models for supporting the actual network operation.**

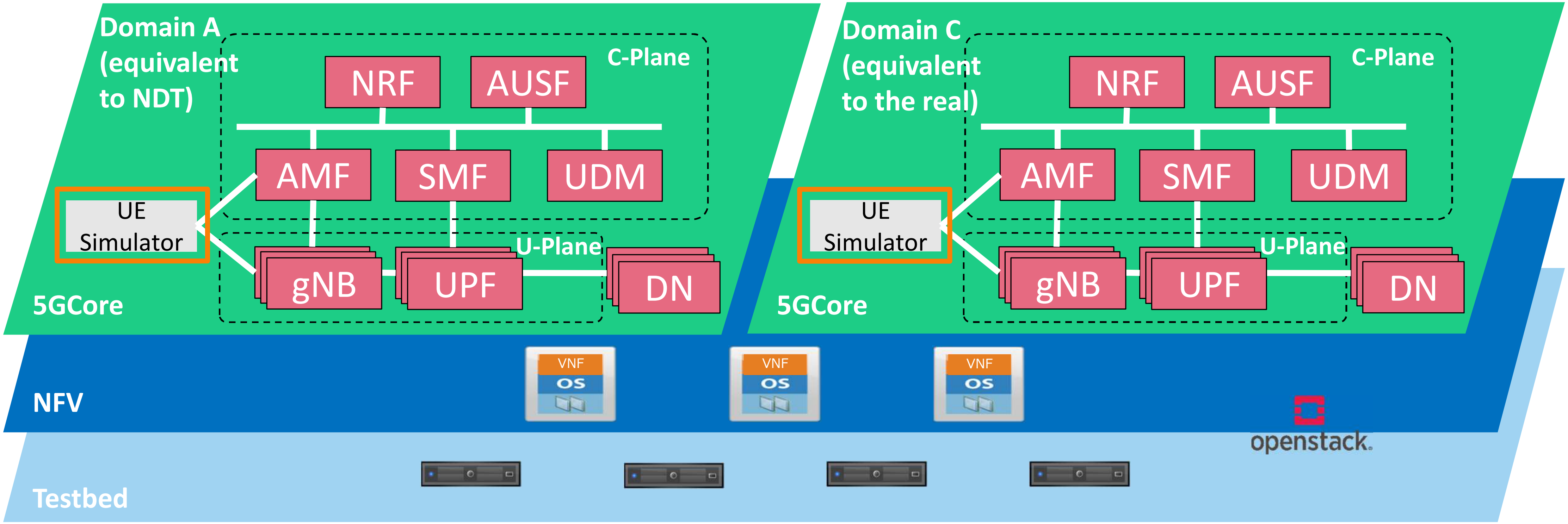
# Purpose of our problem

- To create a good ML model using the dataset generated in network digital twin
  - Network failure classification model, i.e., Root Cause Analysis (RCA) model
  - Train models on training data collected from NDT (Domain A, large volume) and training data from Real (Domain C, small volume)
  - Evaluate the trained models with test data from Real (Domain C)
  - The goal is to develop a good failure classification model in terms of F1-score



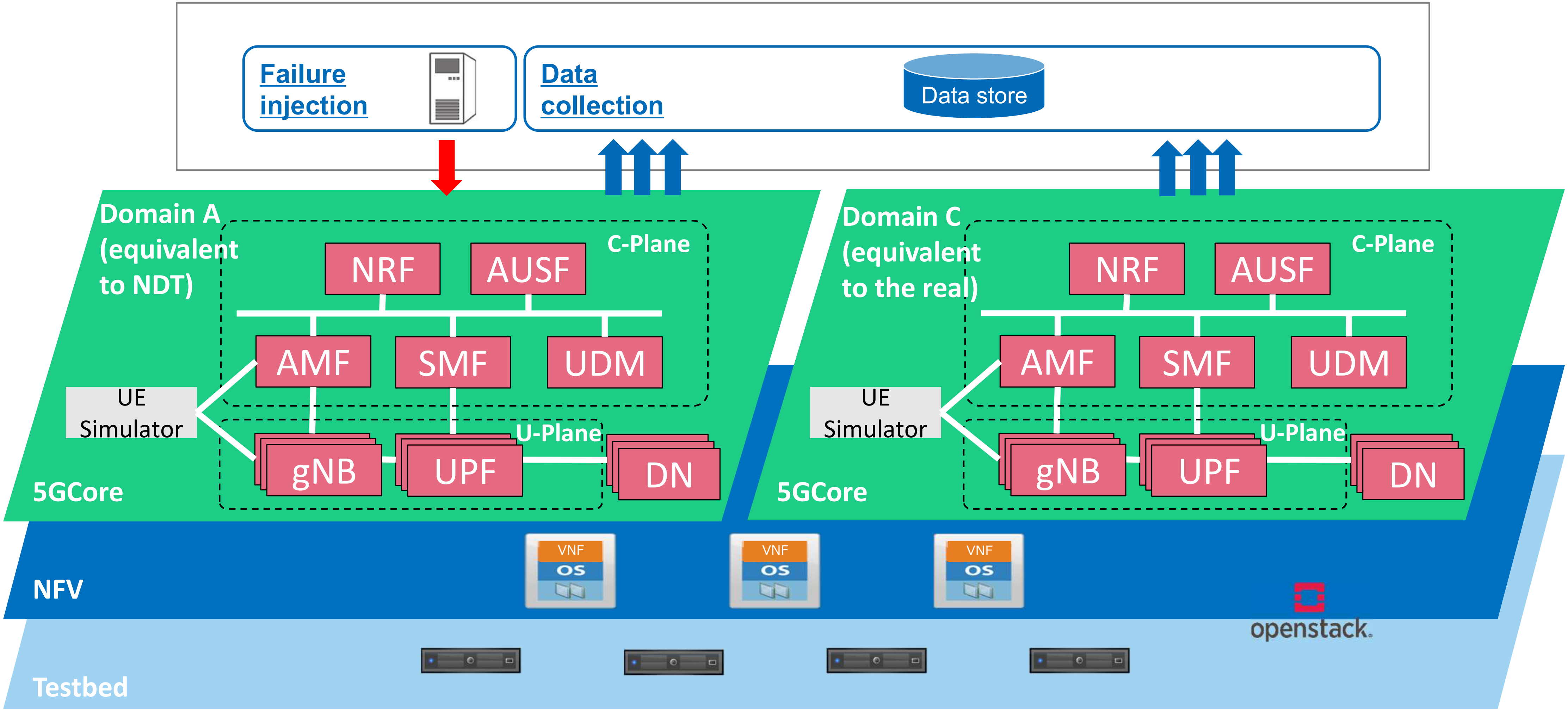
# Environment

We set two VNF-based 5G mobile core networks (same topology)

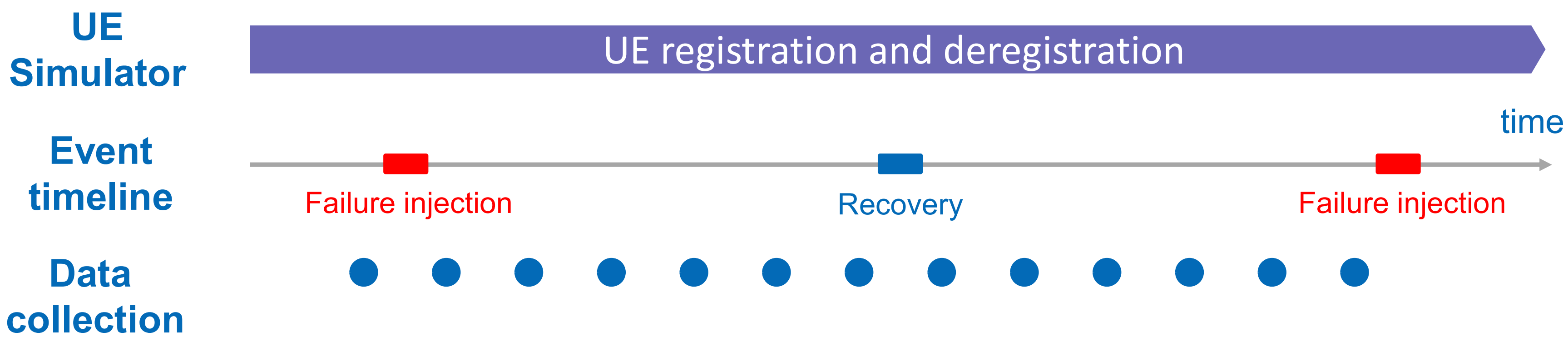


UE Simulator generates different trend of calls

# Environment



# Test scenario



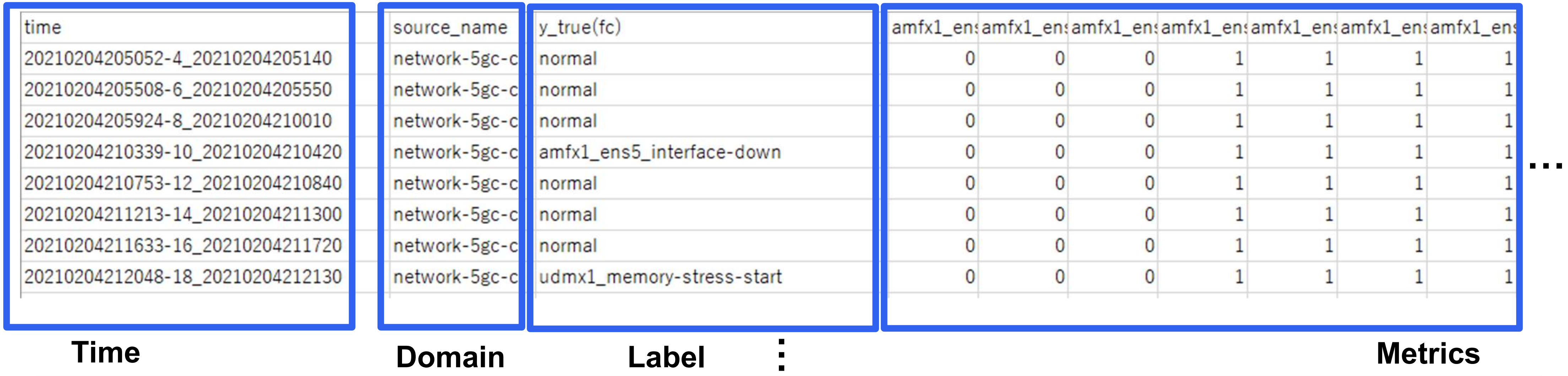
## ■ Failure Event Type

- Failure Type
  - Bridge down (bridge-delif)
  - Interface down (interface-down)
  - Packet loss (interface-loss-start-70)
  - Memory stress (memory-stress-start)
  - CPU overload (vcpu-overload-start)
- Failure Pont
  - AMF (amfx1)
  - AUSF (ausfx1)
  - UDM (udmx1)

- Scenario for domain A and scenario for domain C includes the same failure event type with the same occurrence rate, but total number of failure injections in domain C scenario is smaller than that in domain A scenario.

# Dataset

- For training
  - training-data_a.csv
    - Collected in domain A
    - 80 samples per failure label
  - training-data_c.csv
    - Collected in domain C
    - 10 samples per failure label
- For test
  - test-data_c.csv
    - Collected in domain C
- Dataset Example

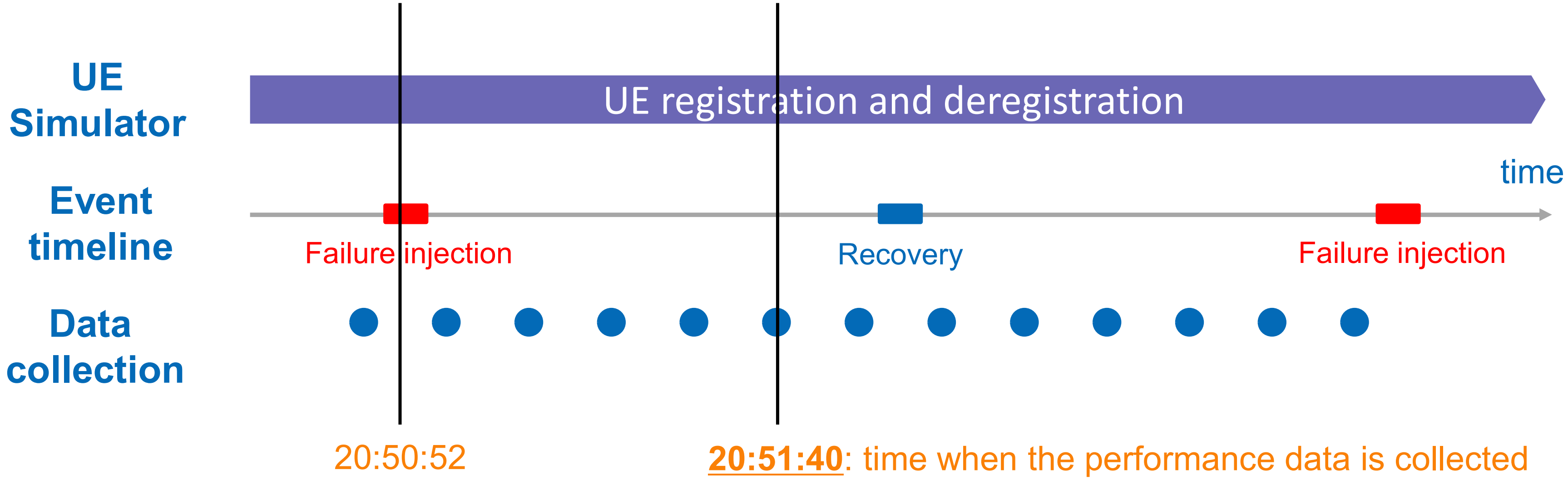


| time | source_name | y_true(fc) | amfx1_ens | amfx1_ens | amfx1_ens | amfx1_ens | amfx1_ens | amfx1_ens | amfx1_ens |
|---|---|---|---|---|---|---|---|---|---|
| 20210204205052-4_20210204205140 | network-5gc-c | normal | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 20210204205508-6_20210204205550 | network-5gc-c | normal | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 20210204205924-8_20210204210010 | network-5gc-c | normal | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 20210204210339-10_20210204210420 | network-5gc-c | amfx1_ens5_interface-down | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 20210204210753-12_20210204210840 | network-5gc-c | normal | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 20210204211213-14_20210204211300 | network-5gc-c | normal | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 20210204211633-16_20210204211720 | network-5gc-c | normal | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 20210204212048-18_20210204212130 | network-5gc-c | udmx1_memory-stress-start | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

# Dataset: Time

## ■ Time

- {start time of event}-{number}_{data collection time}
- Example: “*20210204205052-4_20210204205140*”



# Dataset: Label

- ■ Label for failure classification
  - ■ normal case
    - ●*“normal”*
  - ■ Failure case
    - ●combination of failure point and type
    - ●{failure-point}_{failure-type}
    - ●Example: *“amfx1_ens5_interface-down”*<br>*“udmx1_vcpu-overload-start”*
  - ● Total 16 labels are contained in all datasets.

# Dataset: Metric

## ■ Metric

- {instance name}(_{interface name})_{metric name}_value
  - {interface name} is used only for interface-related metrics

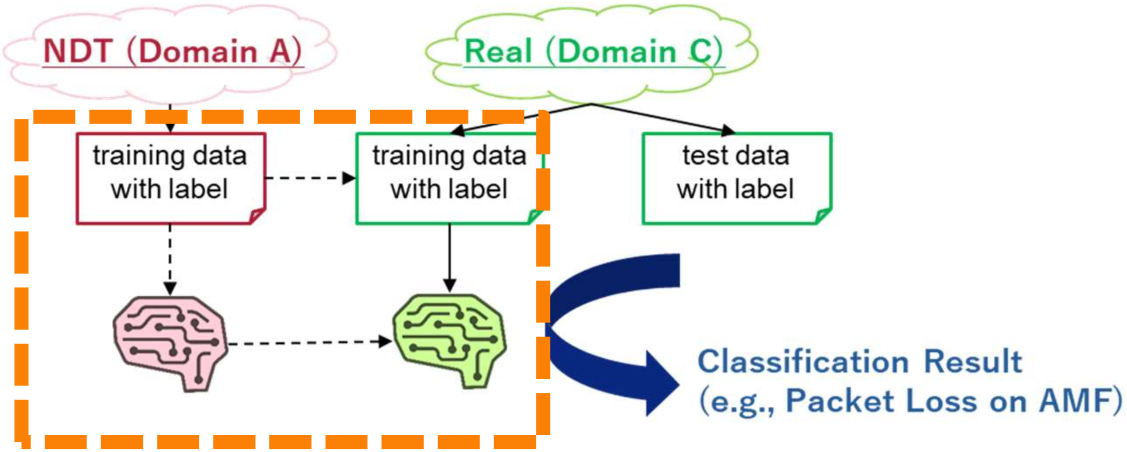
- Interface
  - Example: “*amfx1_ens3_statistics.in-octets_value*”
    - The incoming octets on the network interface “ens3” of instance “amfx1”

- Memory, CPU
  - Example: “*amfx1_memory-stats.used-percent_value*”
    - The percentage of memory usage on the instance “amfx1”

# Task

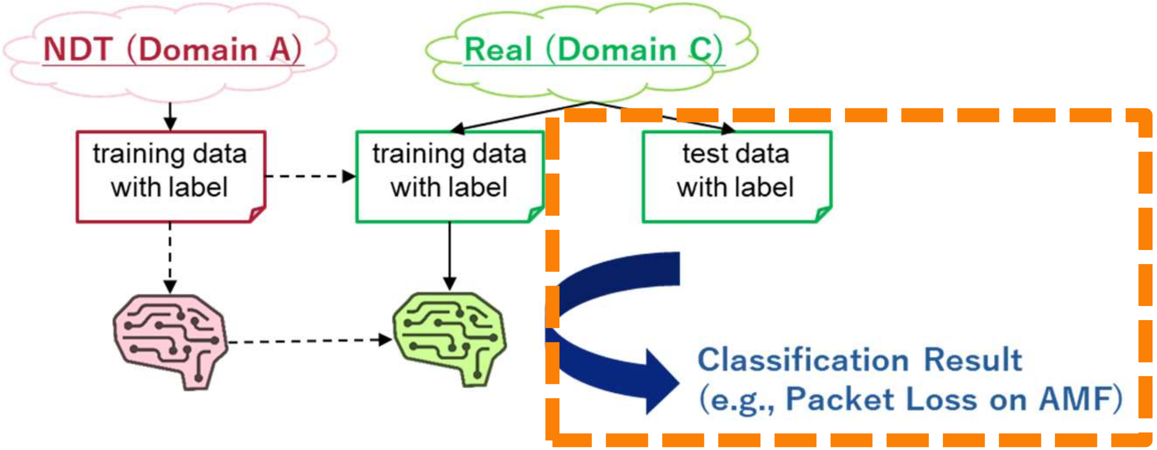
■Create a network failure classification model using the dataset generated in network digital twin

- <u>Training</u>
  - Participants will train a failure classification model for domain C utilizing the training data from domain A and the training data from domain C with label.



# Task

- Create a network failure classification model using the dataset generated in network digital twin
  - Evaluation
    - Participants will evaluate the failure classification result with the test data with label based on the created model.
      1. Create a confusion matrix using the test data
      2. Calculate F1-score for each label



# Task

## ■ Advanced task

- Participants can try advanced task by decreasing the volume of training data from domain C (training-data_c.csv)
- Participants try to develop a good failure classification model for domain C utilizing the training data from domain A and part of the training data from domain C with label.
- This case provides limited training data from the target domain (domain C), therefore more challenging task.

- Training
  1. Participants decrease the training data from domain C.
     Possible number of samples per failure label: 1-9
  2. Participants will train a failure classification model for domain C utilizing the training data from domain A and part of the training data from domain C with label.
- Evaluation
  - Participants will evaluate the failure classification result with the test data with label based on the created model.
  1. Create a confusion matrix using the test data
  2. Calculate F1-score for each label

# Submission

- ■Please submit presentation file containing
  - ●your solution (type of ML model, originality in pre-processing/training method etc.)
  - ●evaluation result including confusion matrix and F1-score of all 16 labels
- ■Participants can include multiple results that are obtained with different volume of domain C training data.

Contact information

aiml-challenge@list.kddi-research.jp