Building a Digital Twin Network

Pere Barlet
(pere.barlet@upc.edu)

Joint work with: Paul Almasan, Guillermo Bernárdez,
Miquel Ferriol, Jordi Paillissé, José Suárez-Varela, Albert Cabellos

Questions

What is a Digital Twin Network?

What can it be useful for?

How can we build one?

How far are we from building it?

How can we leverage AI/ML for this?

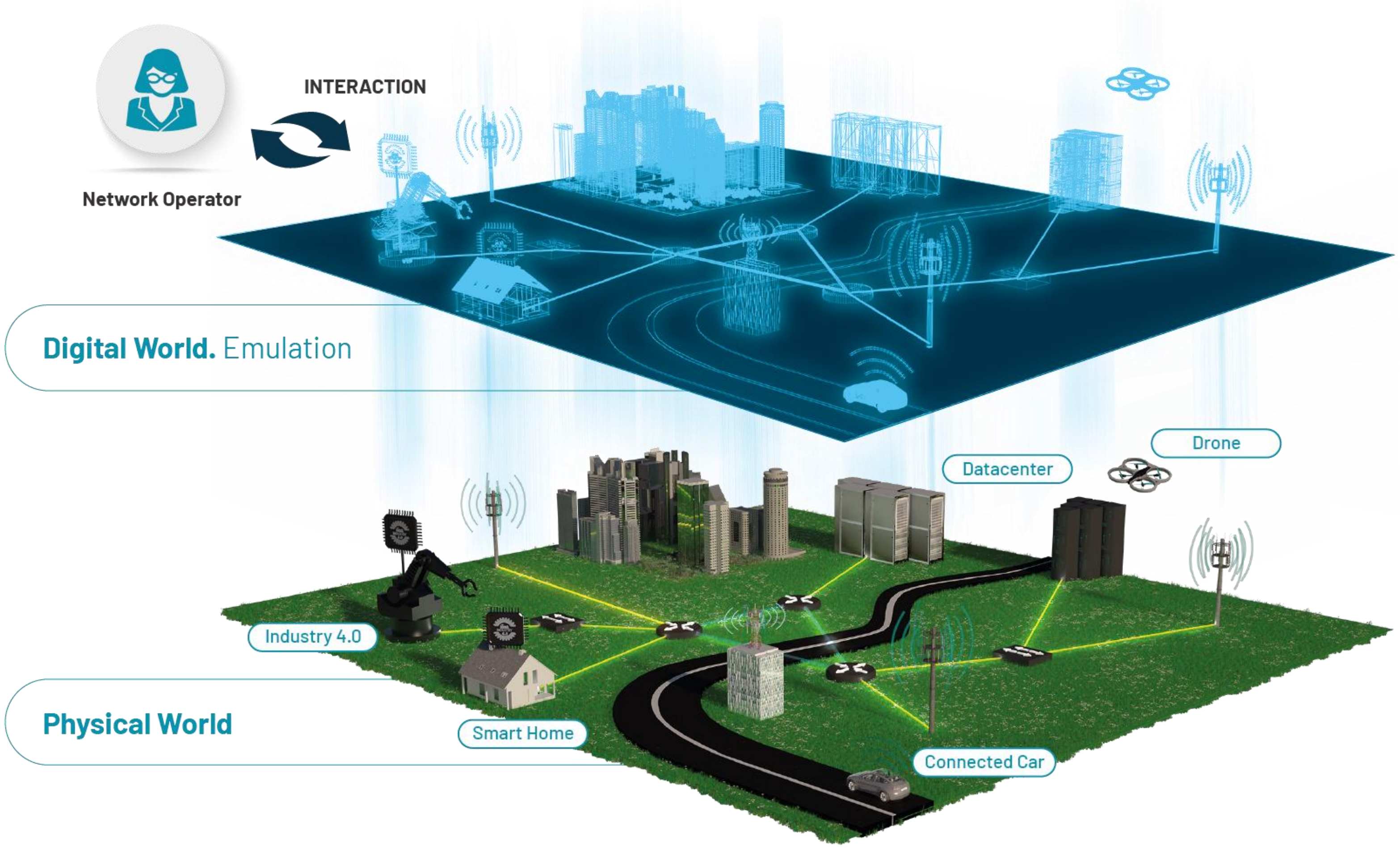
Motivation

- The Internet is a huge, complex and dynamic system
  - >20B connected devices (and growing)[1]
  - 50000 GB/s of data moved daily by the Internet[1]
  - 2.5 Exabytes (10^{18}) of new data created daily[2]

- Digital Twin paradigm
  - Virtual model of a physical object or system represented in the digital world
  - Models complex and dynamic systems
  - Examples: Aerospace, Aeronautics, Industry 4.0, etc.

[1] Cisco Annual Internet Report (2018–2023) White Paper
[2] https://www.ibmbigdatahub.com/infographic/four-vs-big-data

Digital Twin Network (DTN)



- Virtual replica of the communication network
- Model intricacies of real networks in the virtual world
- Test new configurations without compromising the actual network
- IETF, ITU-T working on it[1,2]

[1] C. Zhou et al., "Digital Twin Network: Concepts and Reference Architecture," IETF, Internet-Draft, 2022.

[2] ITU-T, "Digital twin network: Requirements and architecture," Recommendation ITU-T Y.3090, 2022.

BNN-UPC, *Digital Twin Network: Opportunities and Challenges* (2022). https://arxiv.org/abs/2201.01144

DTN applications

Network planning

- Topology design
- Network dimensioning

Performance analysis

- Delay, jitter, loss
- SLA prediction
- Estimate operational bounds

What-if analysis

- Failures
- Traffic changes
- Configuration changes

Network optimization

- Traffic Engineering
- Support more SLA with current resources

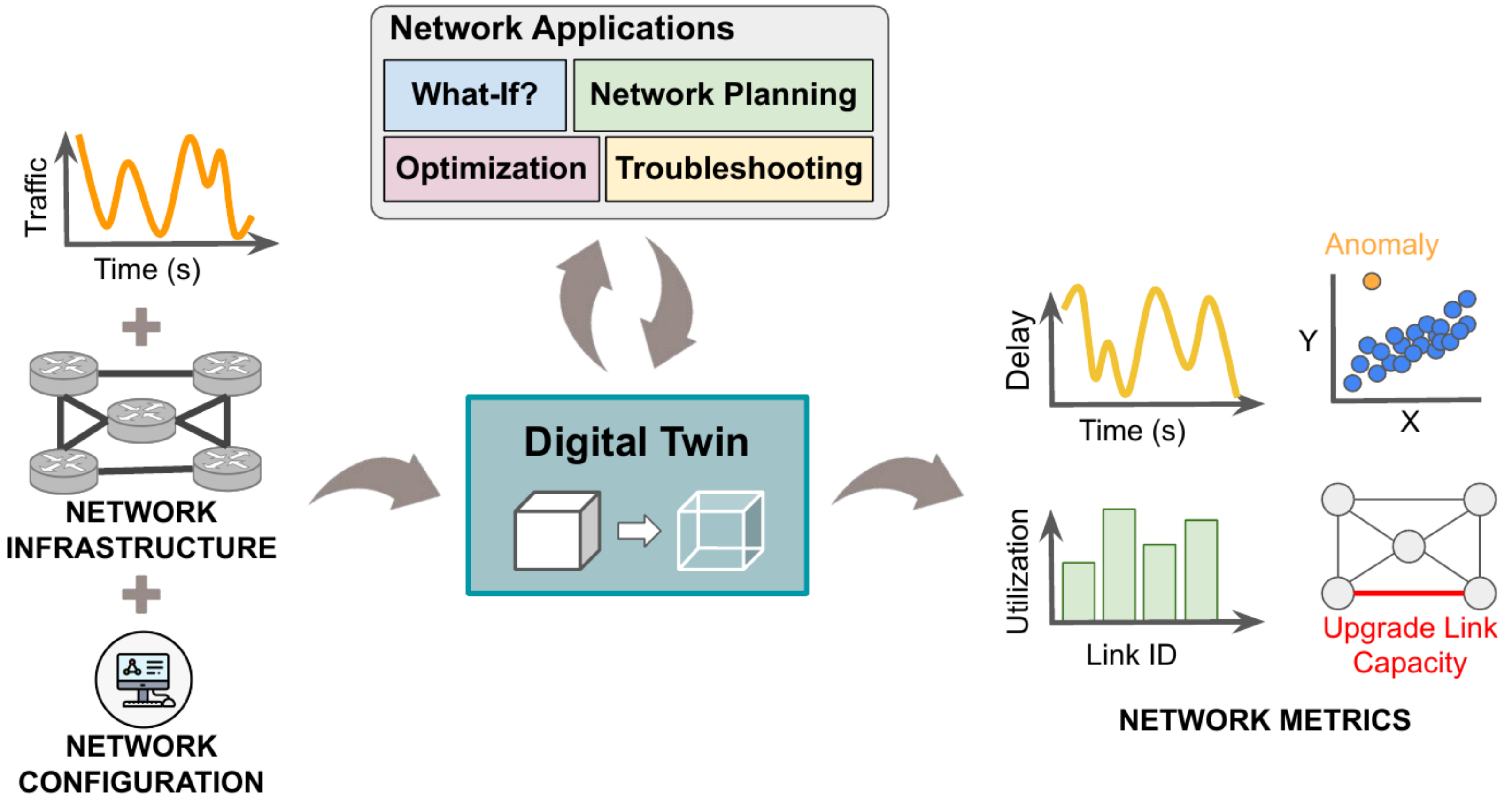
Troubleshooting

- Performance issues
- Anomaly detection

Education and training

- Certifications
- Learning

What is a DTN for us?



A DTN can be as simple or complex as required by the use case

Why does it not exist yet?

Tool	Examples	Cost	Accuracy
Analytical models	Queuing Theory, Fluid models, …	Low	Low
Simulators	NS-3, Omnet++, Packet Tracer, …	High	High
Emulators	GNS3, Mininet, …	High	Low
Testbeds	Fed4FIRE, GENI, …	High	High
Neural networks	MLP, CNN, RNN, …	Low	???

Our research question

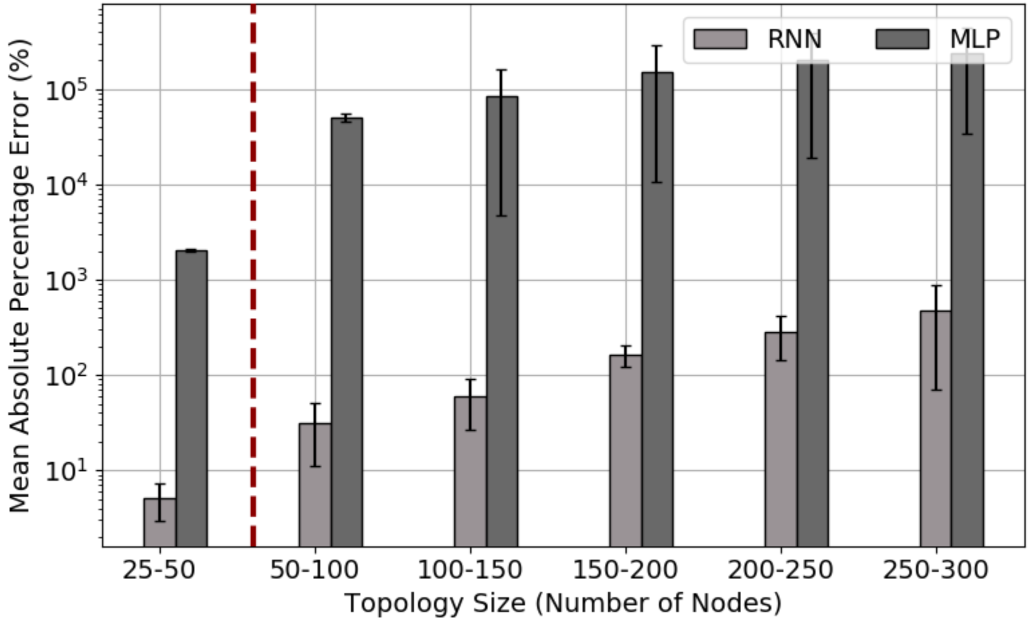
How can we leverage the recent advancements in **AI/ML** to build and **efficient** and **accurate** DTN?

Accurate as network simulators

Efficient as analytical models

Limitations of traditional NN

- Current proposals **do not generalize** to other networks
- Can they be accurate? Yes, but they require training on <u>customer premises</u>
  - Network instrumentation
  - Performance degradation
  - Service disruption



- Why do they not generalize?
  - Networks are essentially graphs (input)
  - Variable size input (nodes/edges)
  - Information is relational

Our vision

(1) Speed and accuracy

- Enables online optimization
- Self-driving networks, closed-loop operation

(2) Generalization

- Operation in networks never seen in training
- Enables instant deployment

(3) Scalability

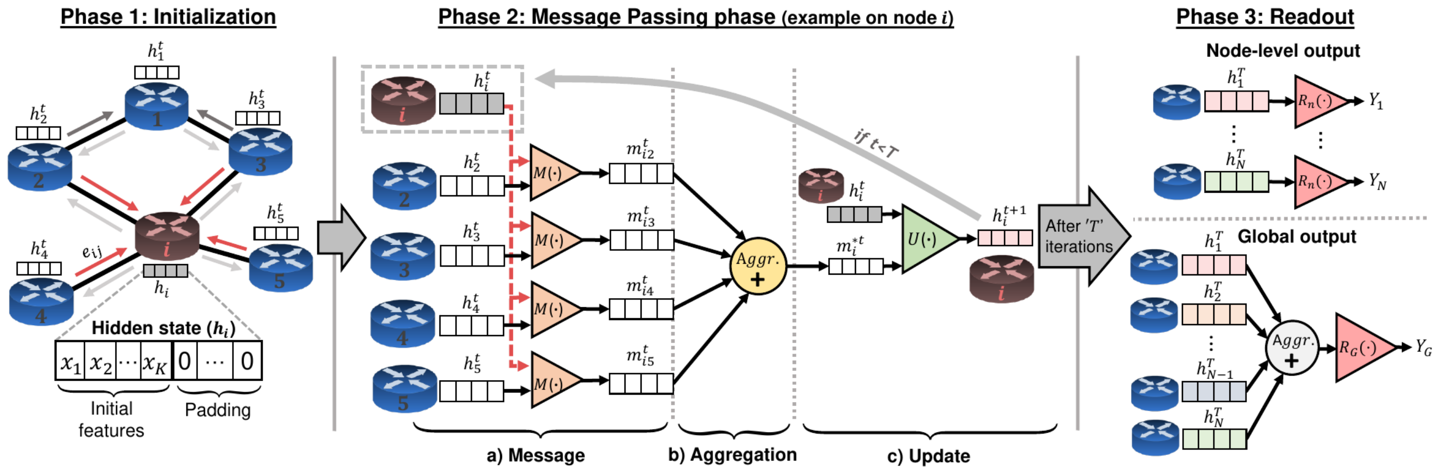
- Scale to *much* larger networks
- Enables training in a controlled testbed

(4) Accessibility

- Accessible to non-ML experts
- Enables easier adoption

Graph Neural Networks

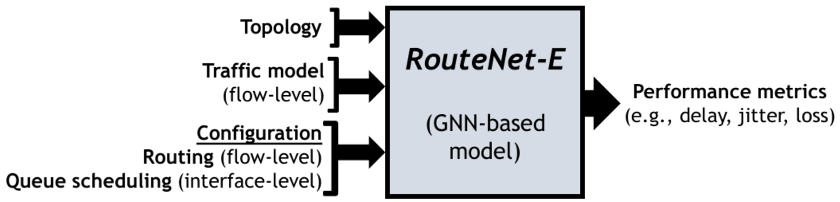
- GNN are a *key* technology to build a DTN because:
  - Assembled dynamically based on the input graph
  - Support variable sized input graphs
  - Model relational information (e.g. graph isomorphism)
  - Generalize to different graphs



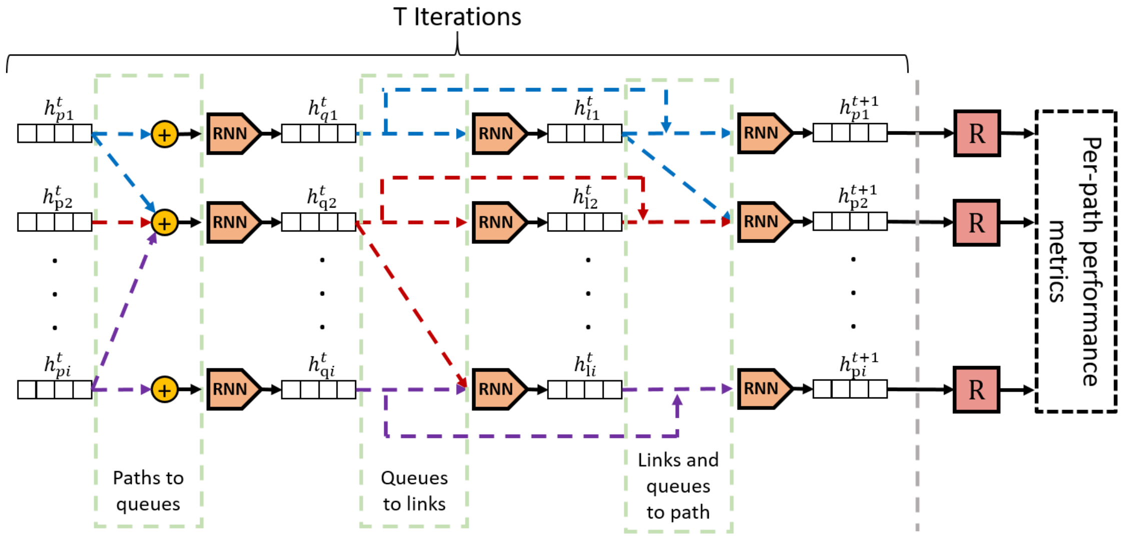
BNN-UPC, *GNN for Communication Networks: Context, Use Cases and Opportunities* (2021). https://arxiv.org/abs/2112.14792

RouteNet

- RouteNet is the reference GNN for network modeling
  - It can operate in networks never seen in training
  - Networks several orders of magnitude larger
  - Cost equivalent to analytical models
  - Accuracy equivalent to simulation

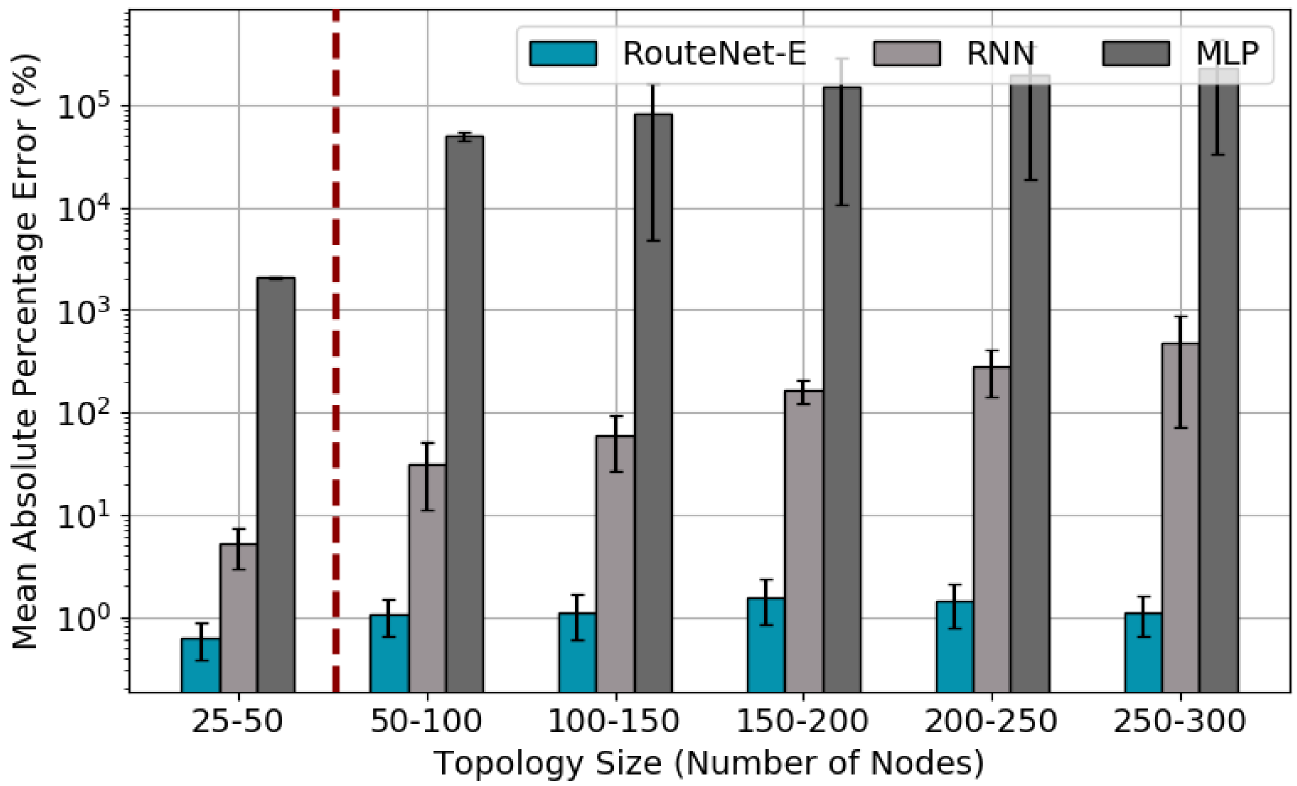


Currently at version E
RouteNet-E (RouteNet-Erlang)

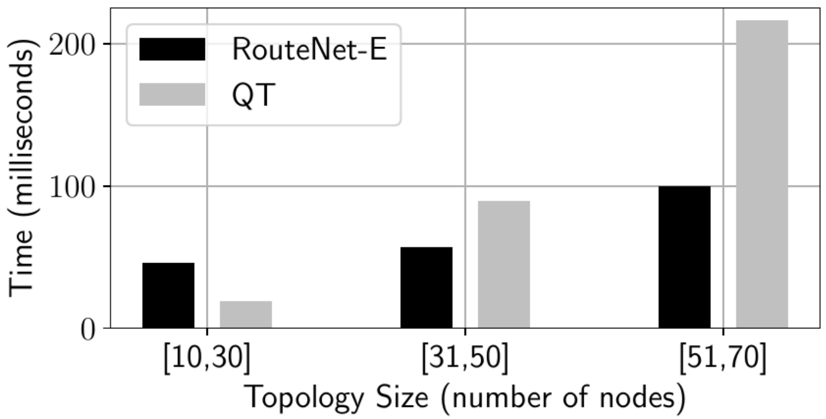


BNN-UPC, *RouteNet-Erlang: A Graph Neural Network for Network Performance Evaluation*. IEEE INFOCOM (2022). https://arxiv.org/abs/2202.13956

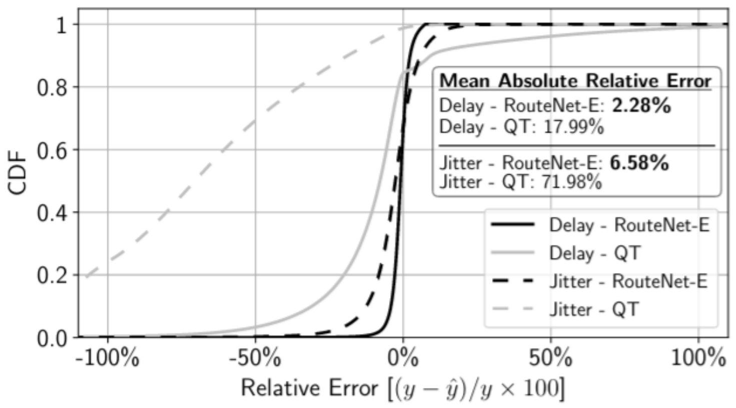
RouteNet-E: Performance



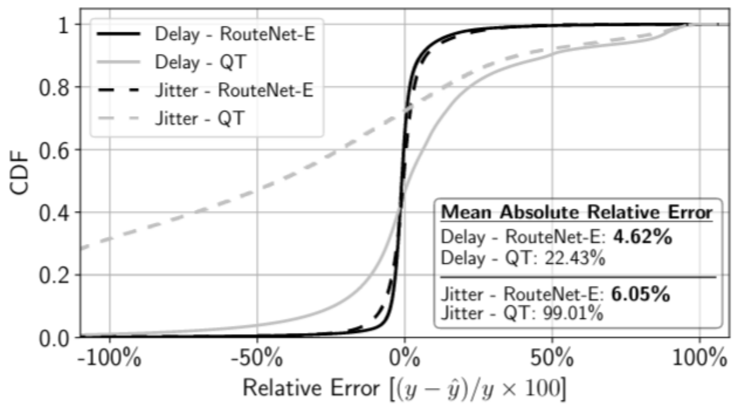
Experiment setup:
- Training: 100K samples, 25-50 nodes simultaneously
- Test: 500 samples 50-300 nodes uniformly



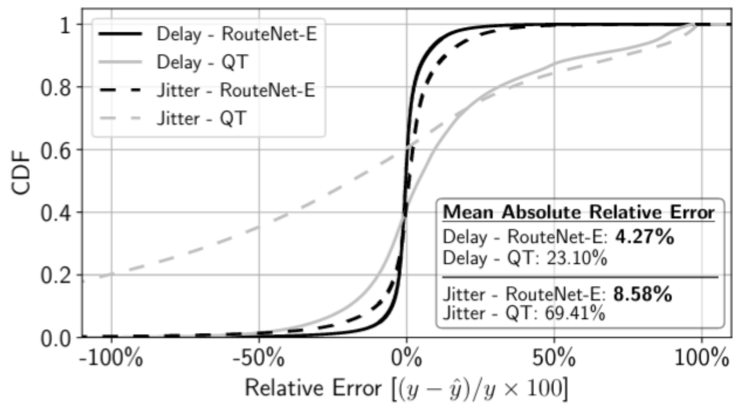
RouteNet-E: Performance



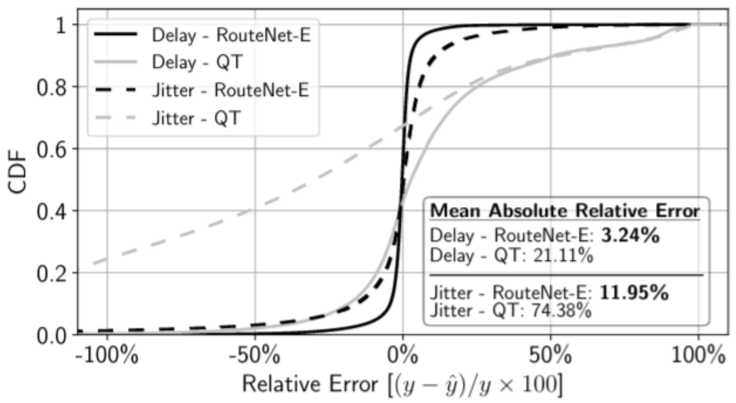
(a) Poisson



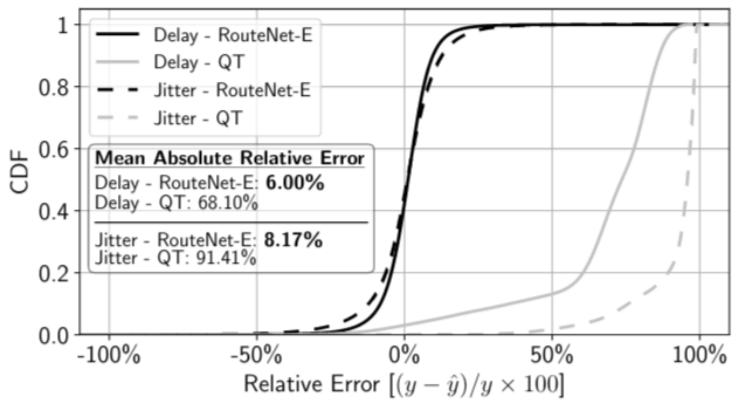
(b) Constant bitrate



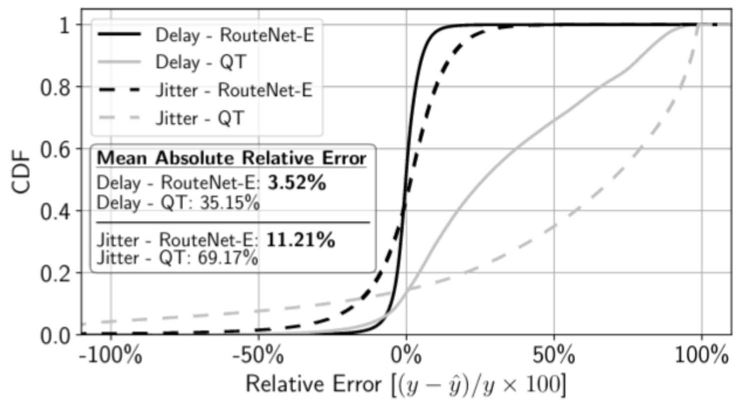
(c) On-Off



(d) Autocorrelated exponentials



(e) Modulated exponentials



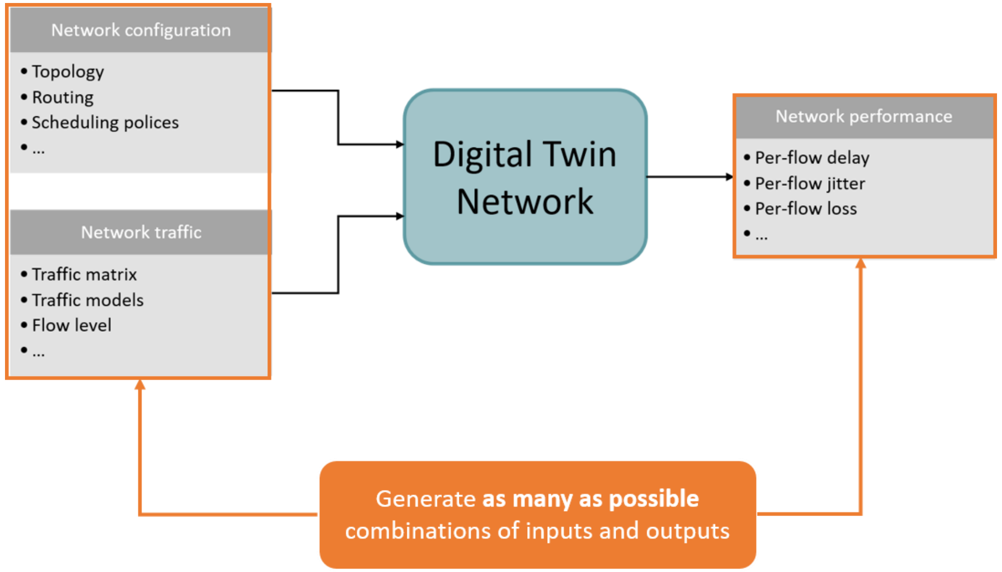
(f) All traffic models multiplexed

BNN-UPC, *RouteNet-Erlang: A Graph Neural Network for Network Performance Evaluation*. IEEE INFOCOM (2022). https://arxiv.org/abs/2202.13956

How to train a DTN?

- Previous ML approaches proposed **online training** (or transfer learning)
  - Not practical from a product point of view
  - Training models is a huge effort, requires a large data set and instrumentation
  - Can lead to degraded performance or service disruption

- RouteNet enables **offline training** in a controlled testbed or simulator
  - Can cover all possible situations
  - Models can be validated before deployment
  - Can provide safety bounds



Conclusions

Tool	Examples	Cost	Accuracy
Analytical models	Queuing Theory, Fluid models, …	Low	Low
Simulators	NS-3, Omnet++, Packet Tracer, …	High	High
Emulators	GNS3, Mininet, …	High	Low
Testbeds	Fed4FIRE, GENI, …	High	High
***Traditional* Neural networks**	MLP, CNN, RNN, …	Low	???

Conclusions

Tool	Examples	Cost	Accuracy
Analytical models	Queuing Theory, Fluid models, …	Low	Low
Simulators	NS-3, Omnet++, Packet Tracer, …	High	High
Emulators	GNS3, Mininet, …	High	Low
Testbeds	Fed4FIRE, GENI, …	High	High
***Traditional* Neural networks**	MLP, CNN, RNN, …	Low	Low

Conclusions

Tool	Examples	Cost	Accuracy
Analytical models	Queuing Theory, Fluid models, …	Low	Low
Simulators	NS-3, Omnet++, Packet Tracer, …	High	High
Emulators	GNS3, Mininet, …	High	Low
Testbeds	Fed4FIRE, GENI, …	High	High
***Traditional* Neural networks**	MLP, CNN, RNN, …	Low	Low
Graph Neural Networks	RouteNet-E	Low	High

Learn: All papers are free online

https://github.com/BNN-UPC/Papers/wiki

Play: Code and Datasets open-source

https://bnn.upc.edu

Code: IGNNITION framework

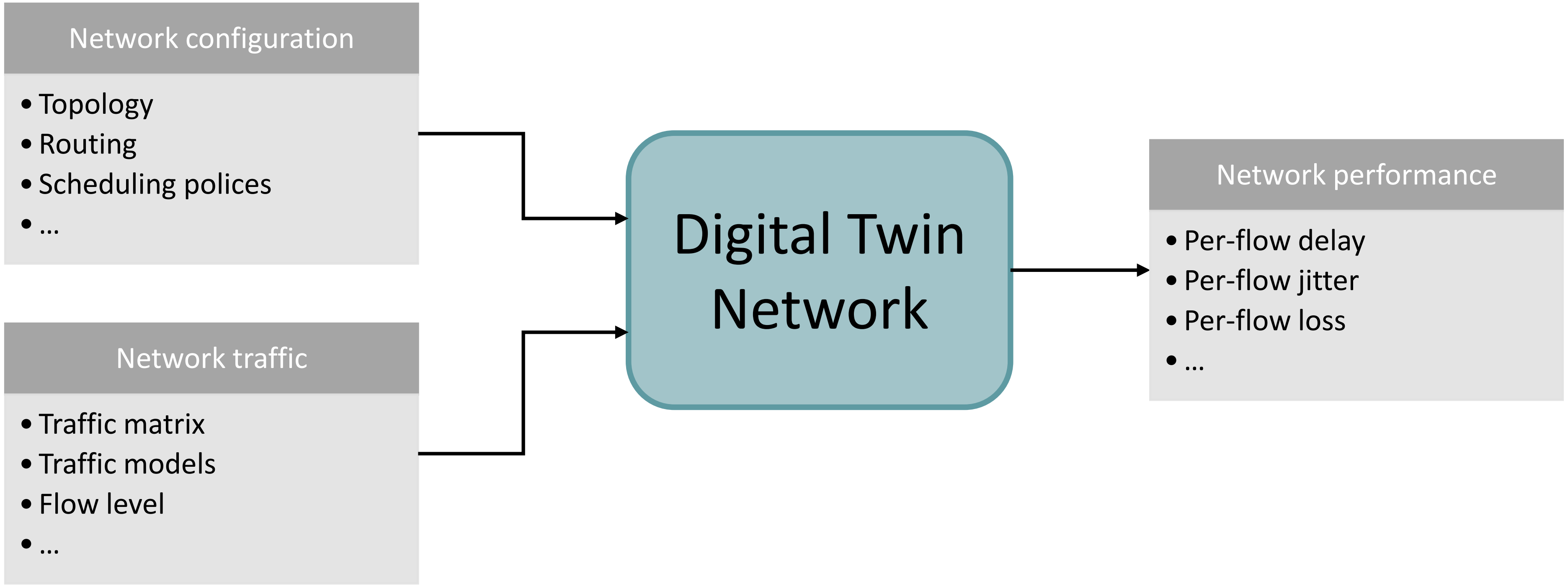
https://ignnition.net

Challenge yourself: GNN Challenge

https://bnn.upc.edu/challenge

Backup slides

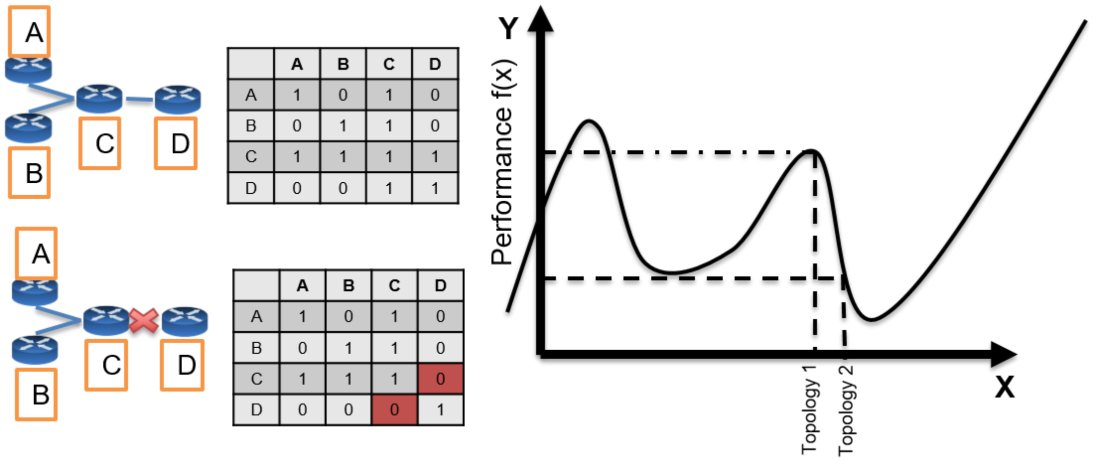
What is a DTN for us?



A DTN can be as simple or complex as needed

Limitations of traditional NN

- Why do they not generalize?
  - Networks are essentially graphs (input data)
  - Graphs are of variable size (links and nodes)
  - Information is relational (as opposed to Euclidian or sequential)
  - Modeling networks with traditional NN is **very hard!**

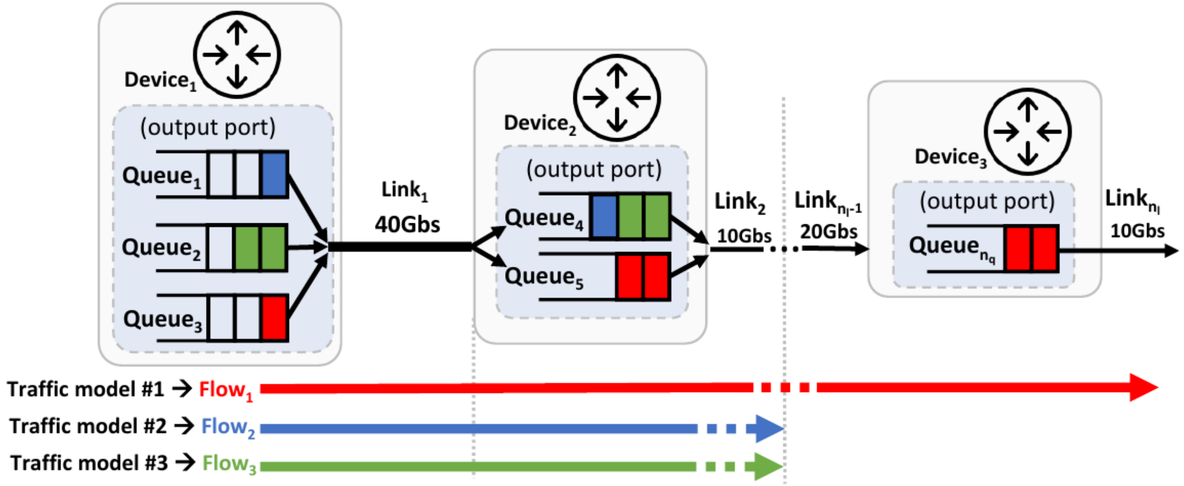


RouteNet-E: Algorithm

Algorithm 1 Internal architecture of RouteNet-E

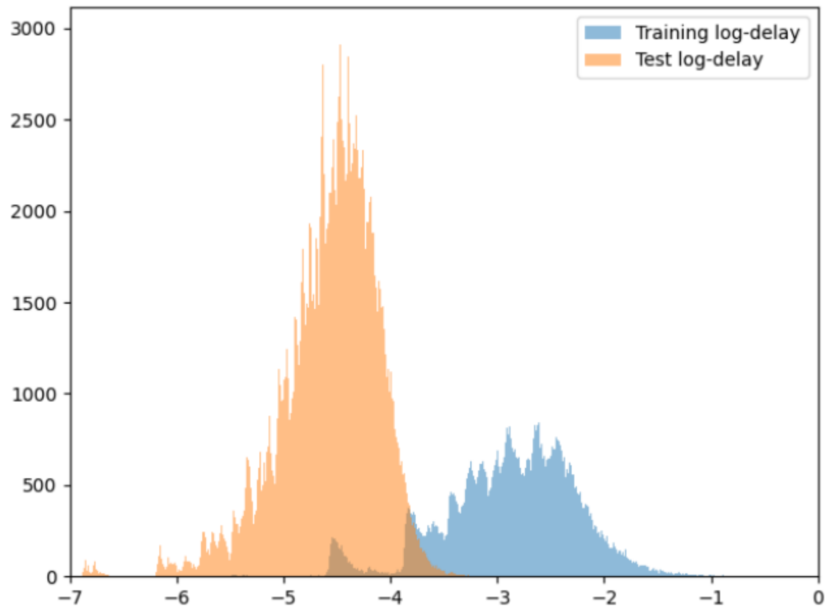
Input: F, Q, L, x_f, x_q, x_l
Output: h_q^T, h_l^T, h_f^T, $\hat{y_f}$, $\hat{y_q}$, $\hat{y_l}$

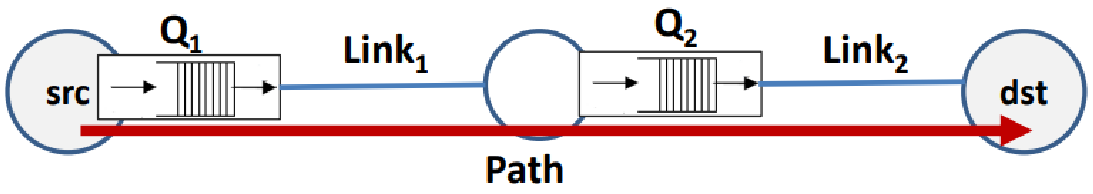
for each $l \in L$ **do** $h_l^0 \leftarrow [x_l, 0...0]$
for each $q \in Q$ **do** $h_q^0 \leftarrow [x_q, 0...0]$
for each $f \in F$ **do** $h_f^0 \leftarrow [x_f, 0...0]$
for t = 0 to T-1 **do** ▷ Message Passing Phase
  for each $f \in F$ **do** ▷ Message Passing on Flows
    for each $(q, l) \in f$ **do**
      $h_f^t \leftarrow FRNN(h_f^t, [h_q^t, h_l^t])$ ▷ Flow: Aggr. and Update
      $\widetilde{m}_{f,q}^{t+1} \leftarrow h_f^t$ ▷ Flow: Message Generation
    $h_f^{t+1} \leftarrow h_f^t$
  for each $q \in Q$ **do** ▷ Message Passing on Queues
    $M_q^{t+1} \leftarrow \sum_{f \in Q_f(q)} \widetilde{m}_{f,q}^{t+1}$ ▷ Queue: Aggregation
    $h_q^{t+1} \leftarrow U_q(h_q^t, M_q^{t+1})$ ▷ Queue: Update
    $\widetilde{m}_q^{t+1} \leftarrow h_q^{t+1}$ ▷ Queue: Message Generation
  for each $l \in L$ **do** ▷ Message Passing on Links
    for each $q \in L_q(l)$ **do**
      $h_l^t \leftarrow LRNN(h_l^t, \widetilde{m}_q^{t+1})$ ▷ Link: Aggr. and Update
    $h_l^{t+1} \leftarrow h_l^t$
$\hat{y_f} \leftarrow R_f(h_f^T)$ ▷ Readout phase
$\hat{y_q} \leftarrow R_q(h_q^T)$



RouteNet-E: Scalabilty

- Out-of-distribution
  - Larger paths
  - Larger link capacities
  - Output distributions (delay)



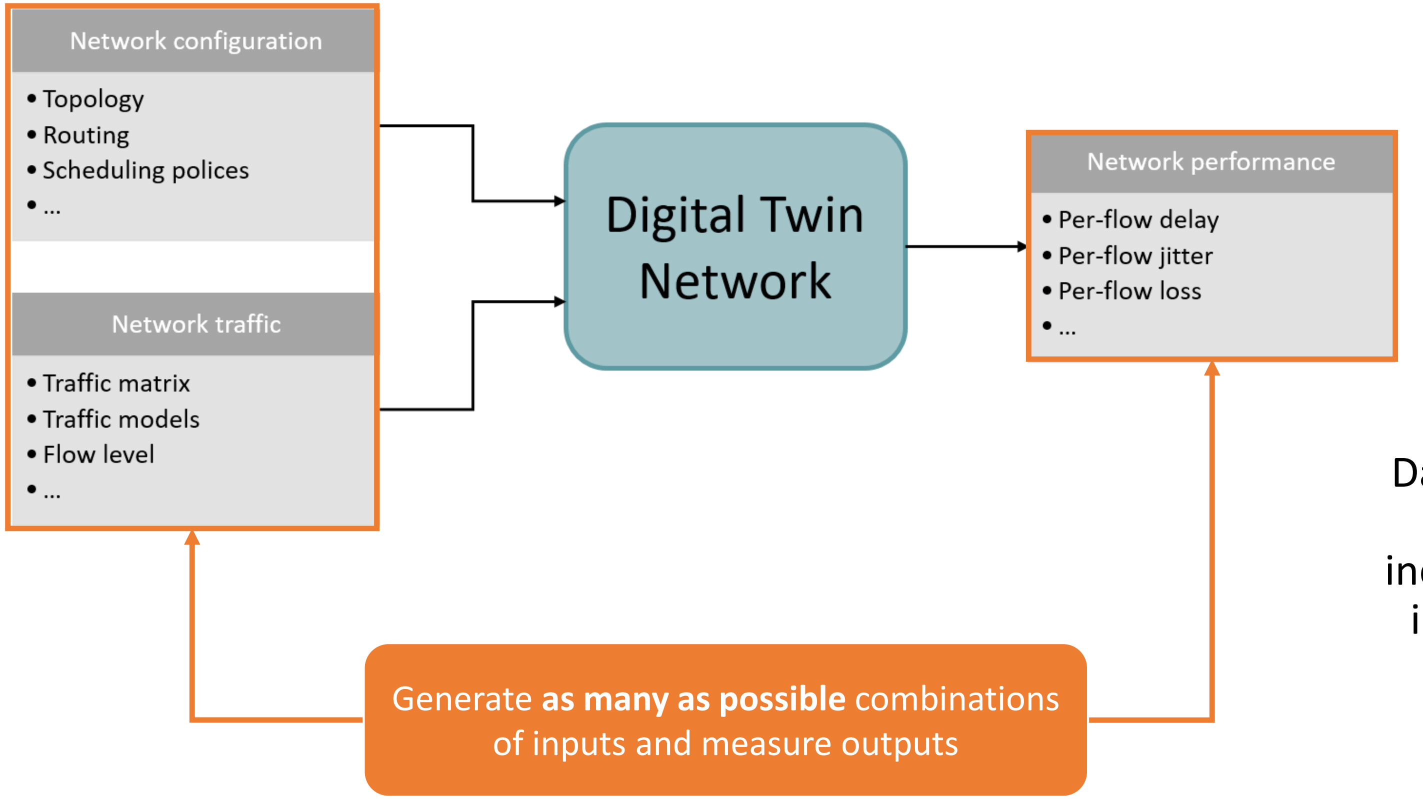


Delay$_{L1}$ = Avg_utilization$_{Q1}$ * Size$_{Q1}$ / Cap$_{L1}$

Delay$_{Path}$ = $\sum_{k=1}^{N\ links} Delay_{L_k}$

Our solution

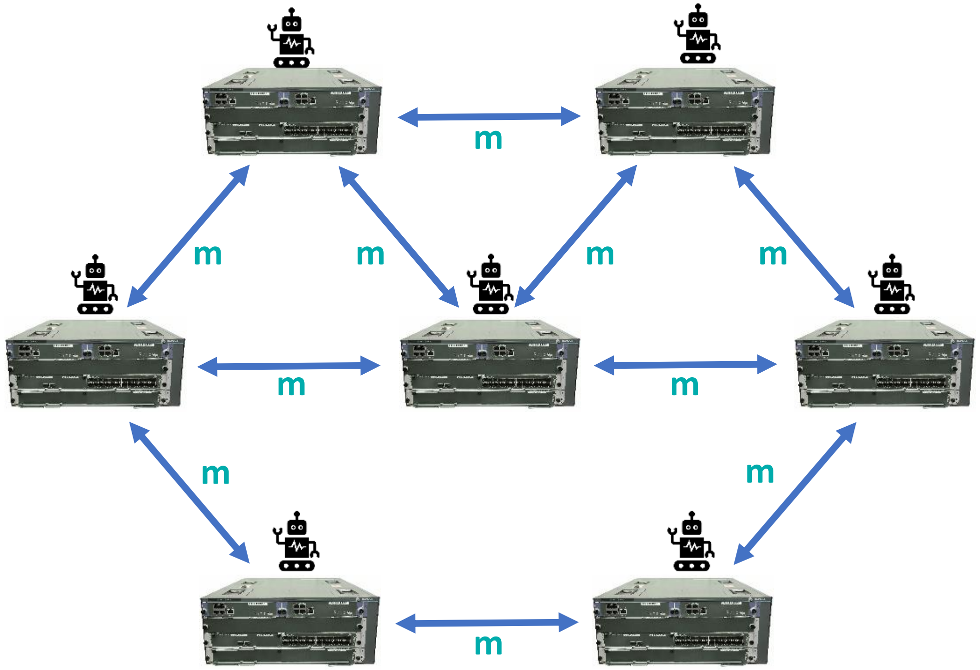
- Small-scale testbed at **vendor** premises (before deployment)
- Leverage GNN generalization/scalability features



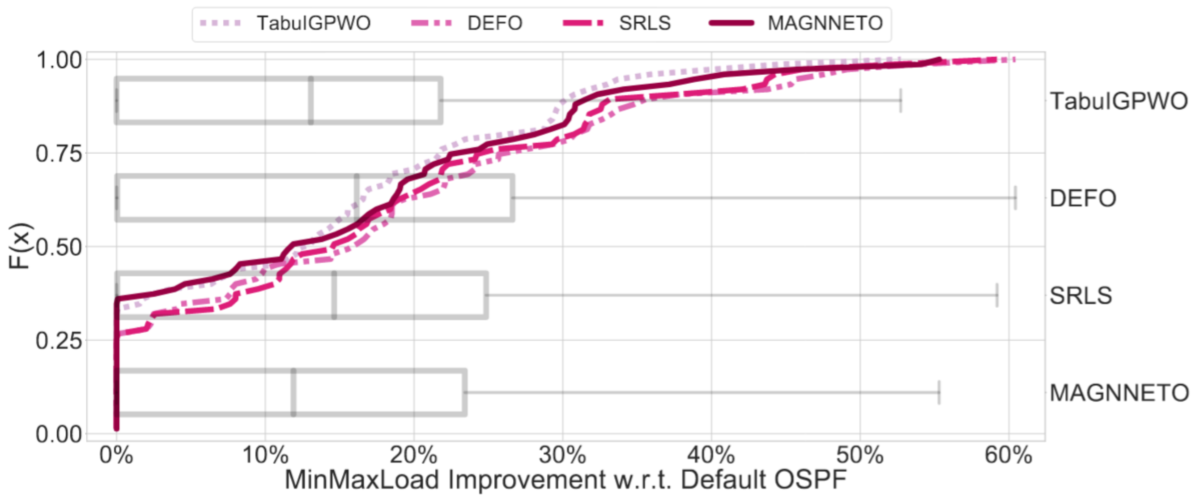
Data sets need to contain all possible scenarios, including those that result in poor performance or service disruption !!

Multi-Agent Distributed Optimization

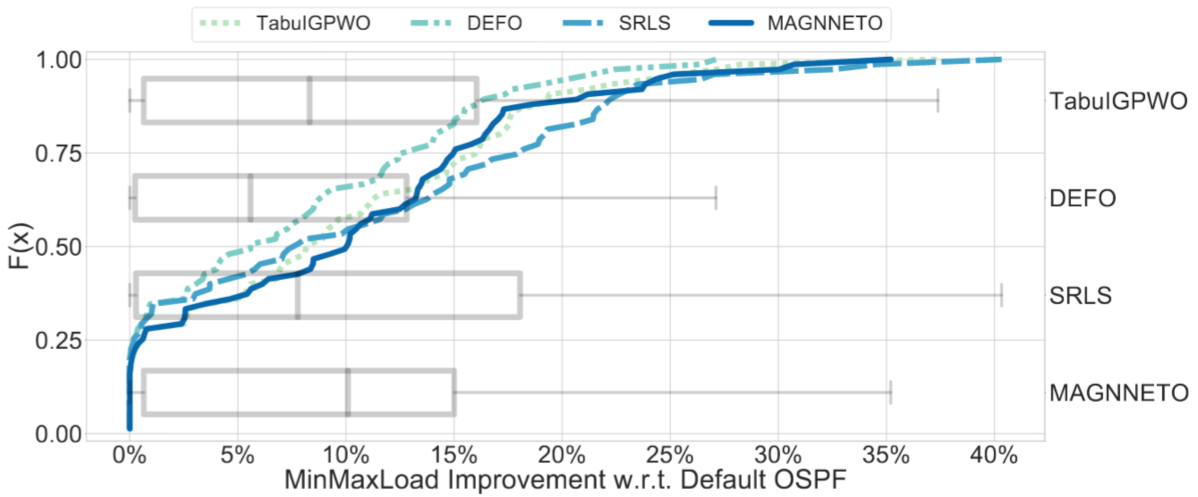
- Agent that operates networks autonomously
- Exchanges messages (protocol) with other agents
- Uses a GNN to learn message contents
- Optimizes network performance using Multi-Agent RL
  - Example: Traffic Optimization, Congestion, etc.
- Scales well as it naturally (GNN) distributes load among devices
- **Supports offline learning, can operate in unseen networks**



MARL+GNN: Optimization



(a) TopologyZoo Uniform



(b) TopologyZoo Gravity

MARL (Proximal Policy Optimization) + GNN
100 TM (uniform / gravity)
75 topologies from Topology Zoo
Trained with 2 topologies (NSFnet, GEANT2)