

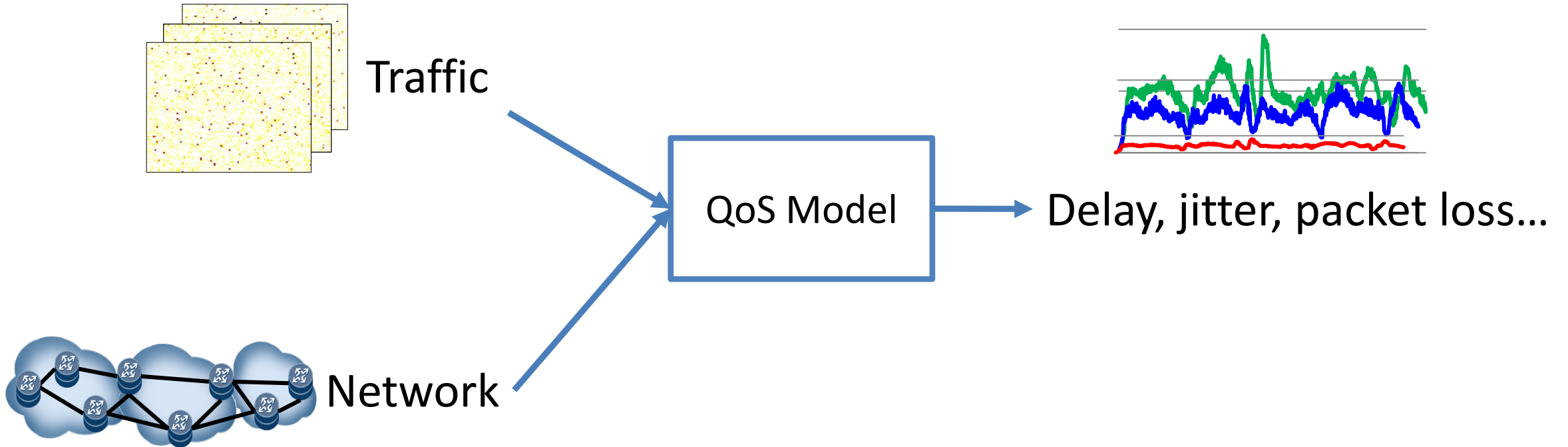
Deep-Q: Traffic-driven QoS Inference using Deep Generative Network

Shihan Xiao, Dongdong He, Zhibo Gong

Network Technology Lab,
Huawei Technologies Co., Ltd., Beijing, China

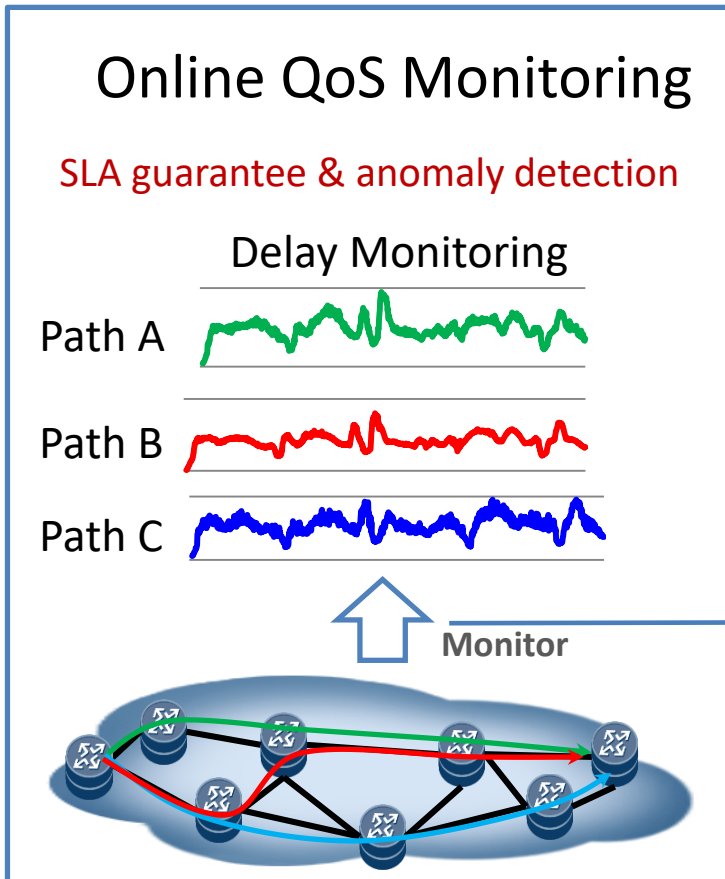
Background

- What is a QoS Model?



Background

- Why is it important?



A QoS model helps **reduce most of the cost!**



Require **high cost** on real-time active QoS measurements!

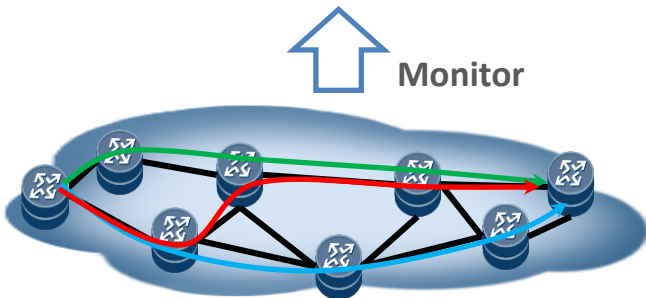
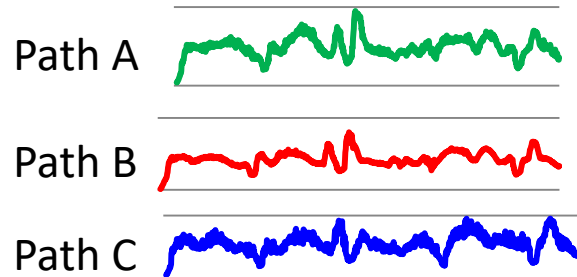
Background

- Why is it important?

Online QoS Monitoring

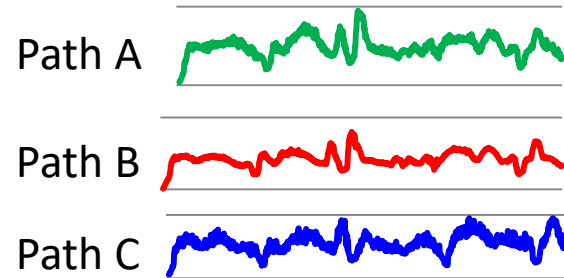
SLA guarantee & anomaly detection

Delay Monitoring



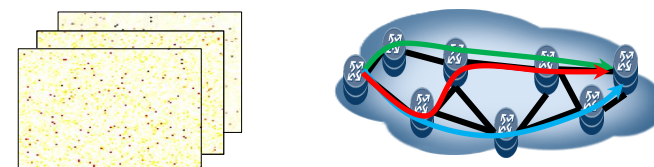
Offline Traffic Analysis

Delay Inference



Inference

Traffic trace + Network



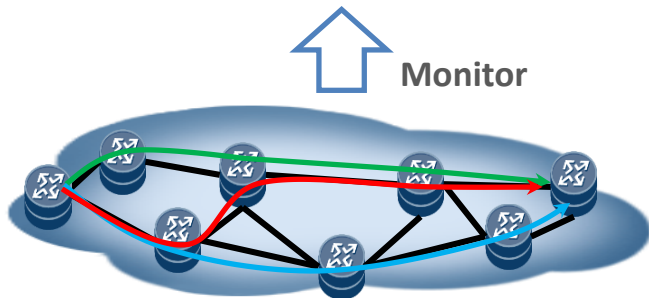
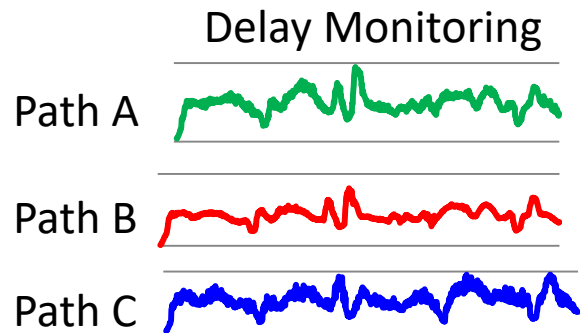
A QoS model can do QoS inference
without QoS measurements

Background

- Why is it important?

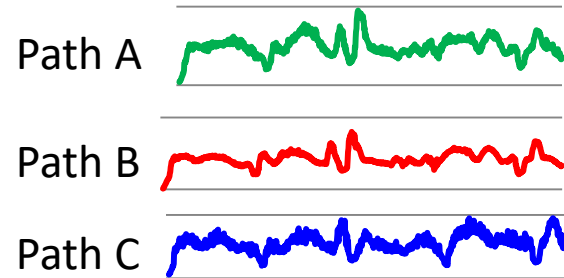
Online QoS Monitoring

SLA guarantee & anomaly detection



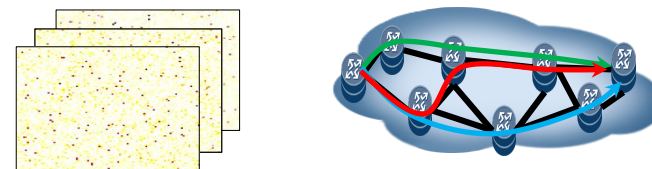
Offline Traffic Analysis

Delay Inference



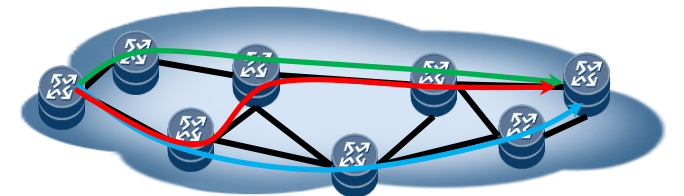
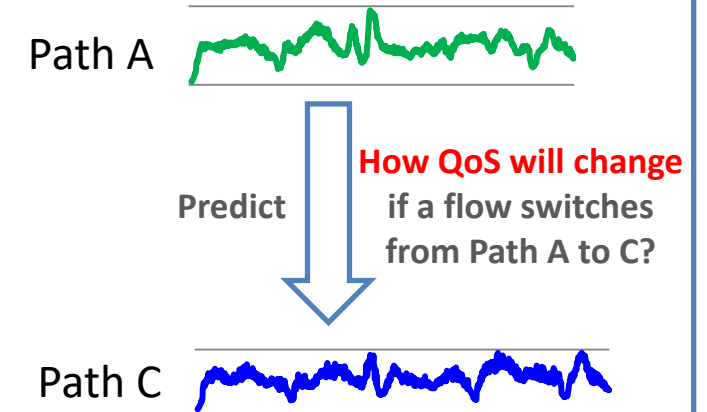
Inference

Traffic trace + Network



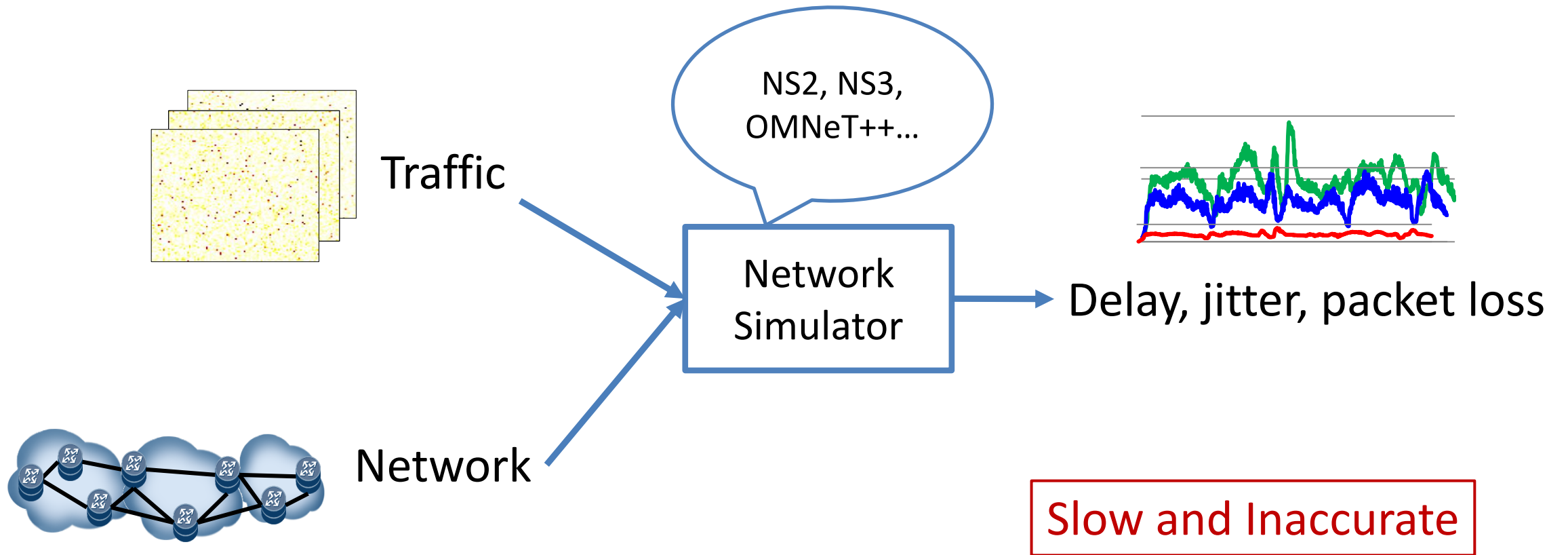
“What if” Analysis

Delay Prediction



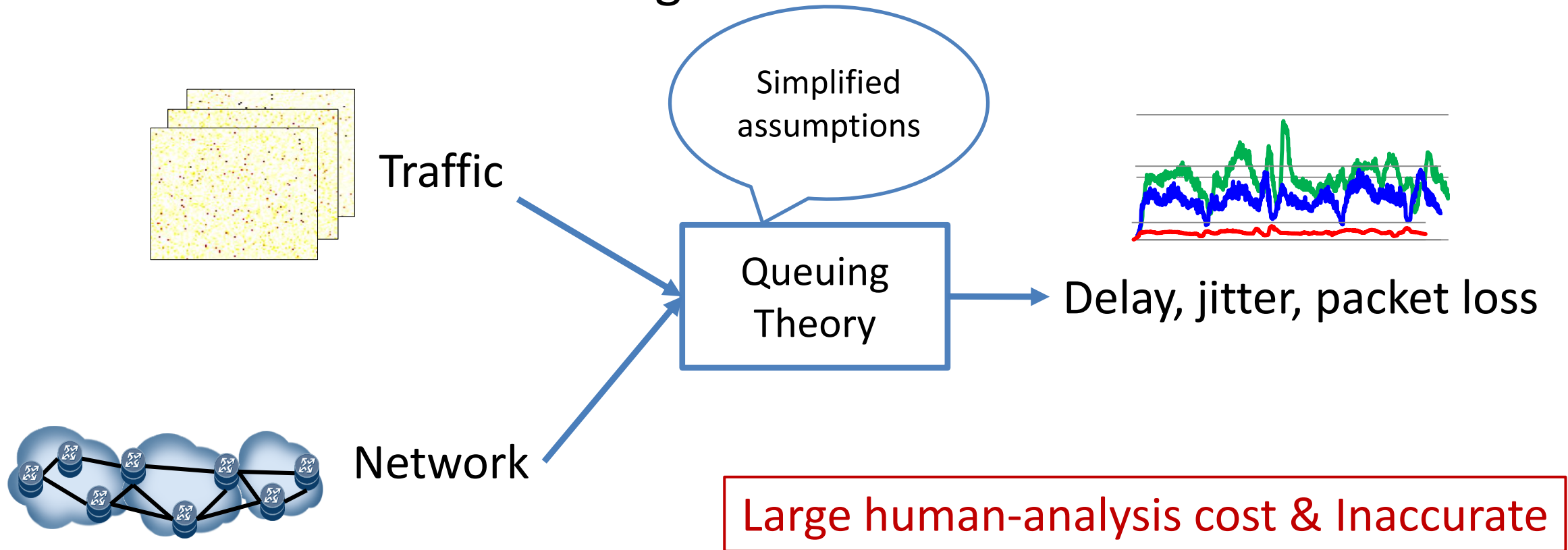
Traditional Methods

- 1. Network simulator



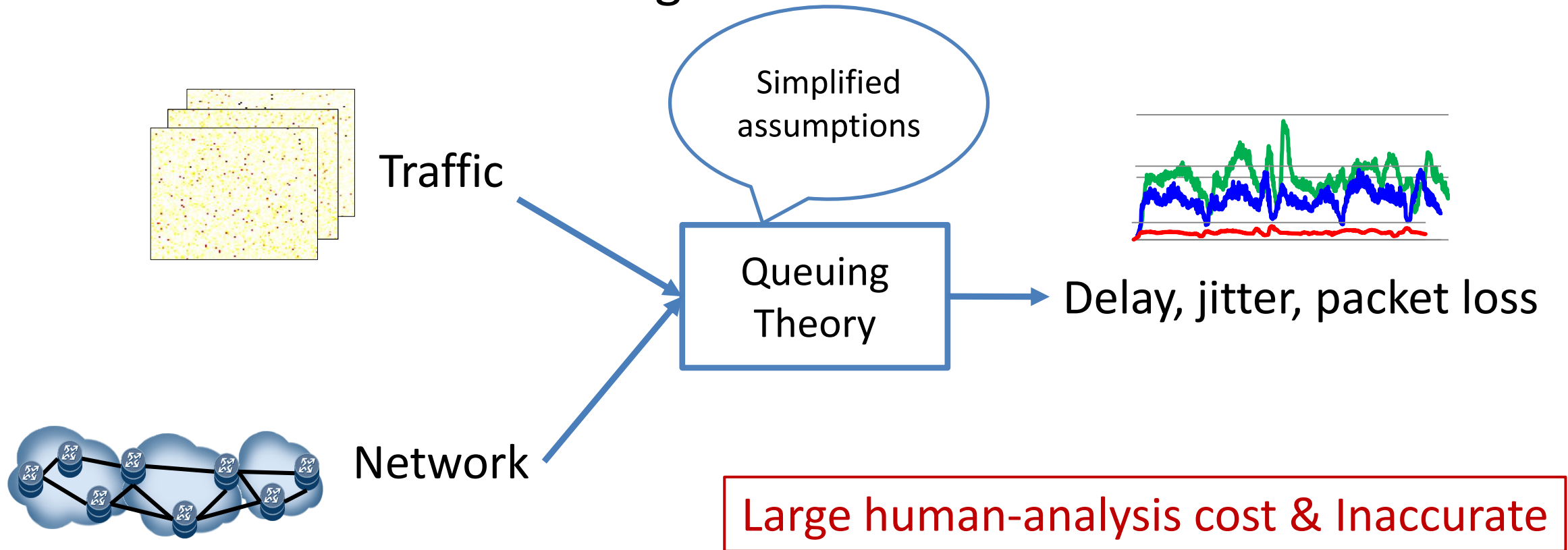
Traditional Methods

- 2. Mathematical modeling



Traditional Methods

- 2. Mathematical modeling



A fast, accurate & low-cost QoS model is helpful!

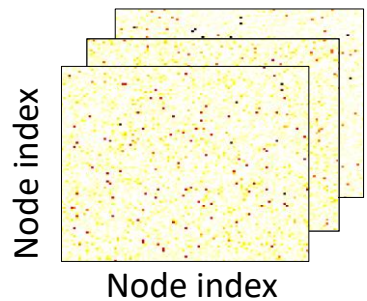
Key Observations

- Observation 1: **Traffic load per link is much easier to collect** & well-supported by existing tools (e.g., SNMP) than QoS values per path

Key Observations

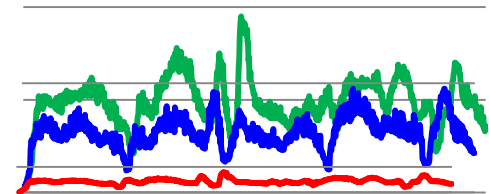
- Observation 1: **Traffic load per link is much easier to collect** & well-supported by existing tools (e.g., SNMP) than QoS values per path
- Observation 2: Traffic load is the key factor of QoS changes

Traffic: collected link load matrixes



QoS Model

Delay, jitter, packet loss



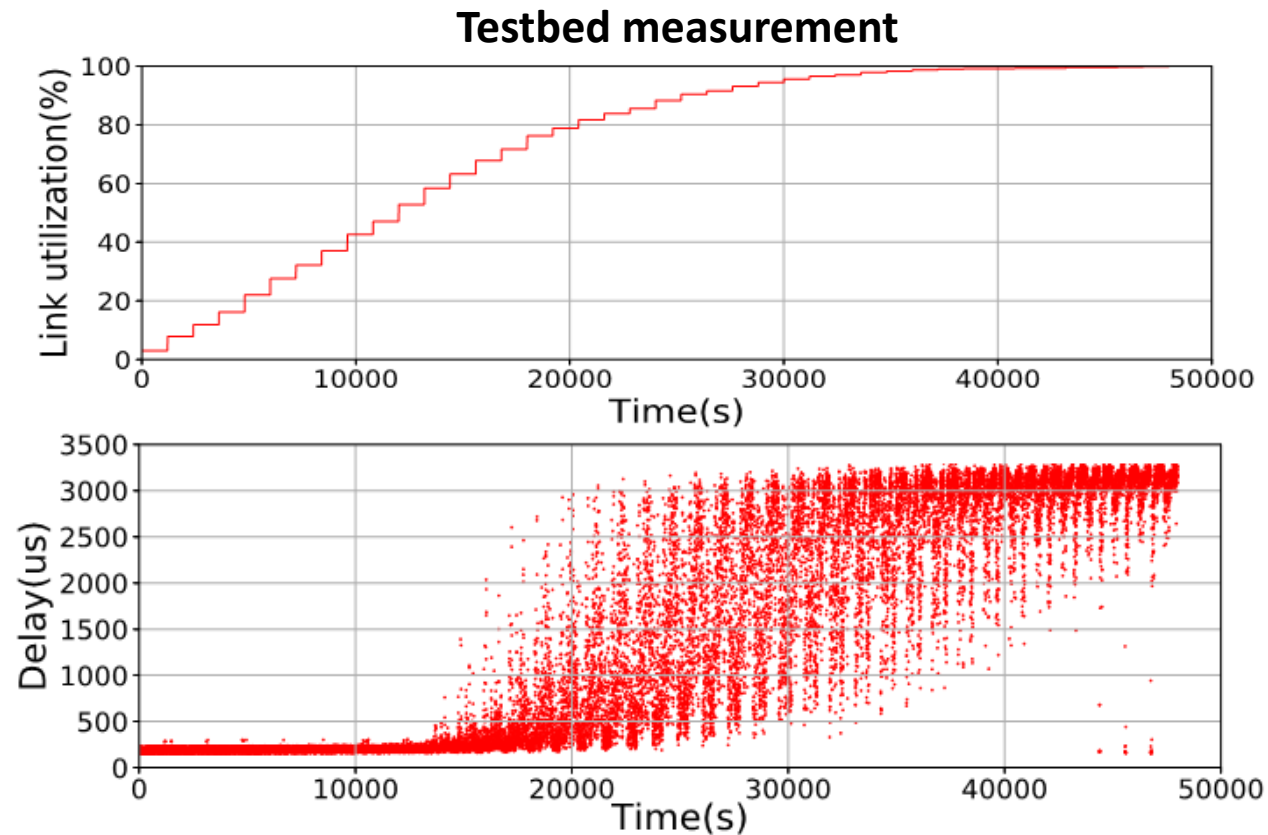
Key Observations

- Observation 3: Different traffic loads lead to different QoS distributions

40 traffic loads (per 20 min)

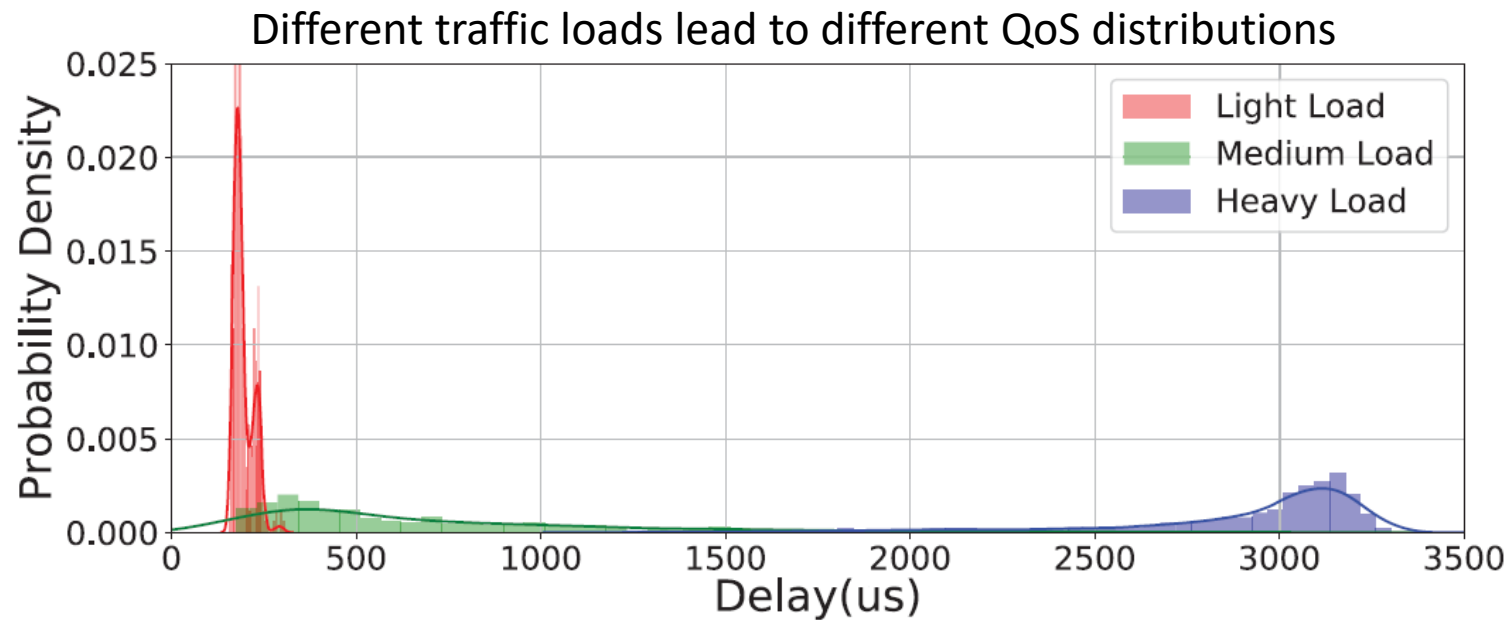


Measured delay samples



Key Observations

- **Target Problem:** Given a set of traffic load matrixes during time T, what are the **distributions of QoS values** (delay, jitter, loss...) of each network path during T?

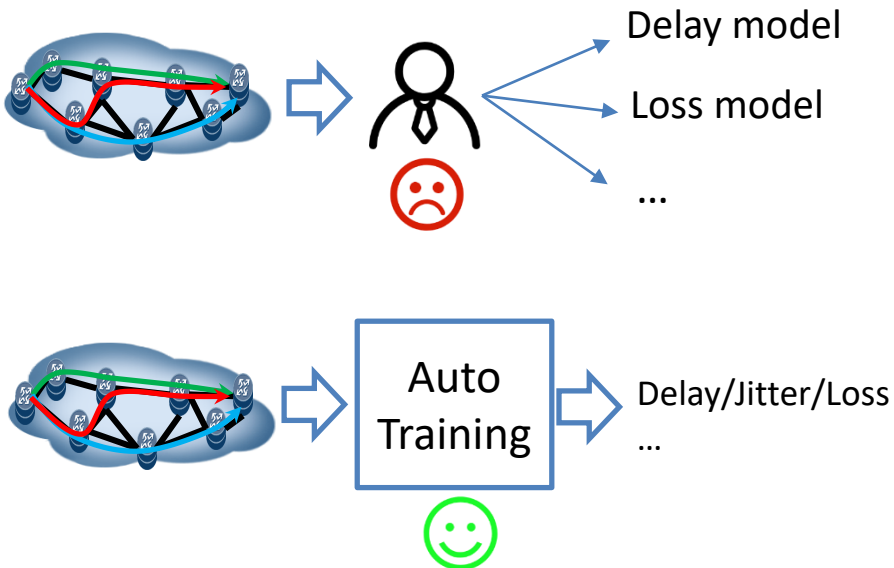


Solution of Deep-Q

- Why deep learning helps?

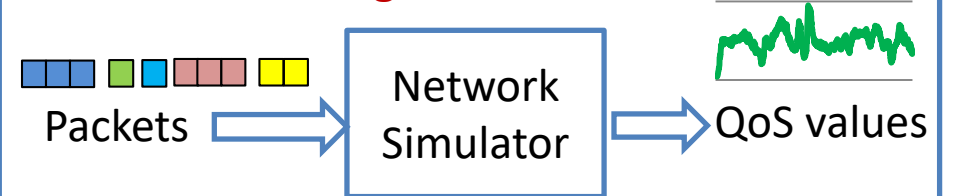
Low human-analysis cost

Data-driven VS. Human-engineered model

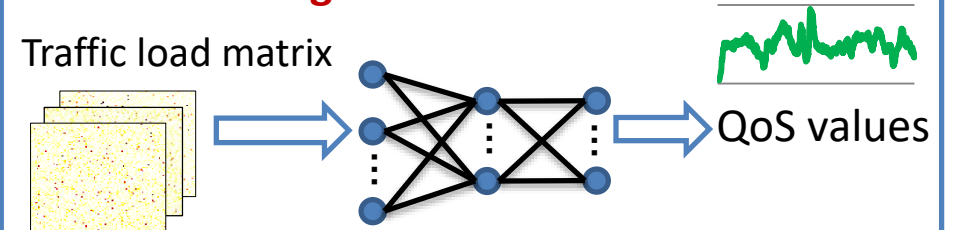


Fast inference

Running time of Hours!



Running time of Milliseconds!



Key Technology: Deep Generative Network

So what is the difference?

- State-of-the-art DGNs in deep learning

Image domain

Network domain

- **GAN**(Generative Adversarial Network) & **VAE**(Variational Autoencoder)

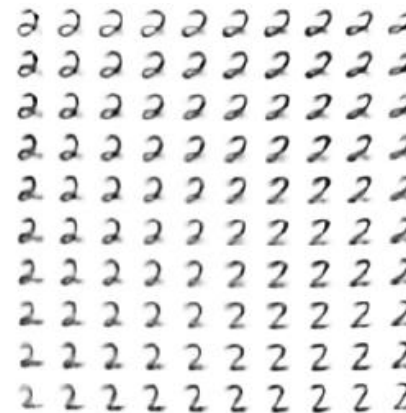
Input: “this small bird has a pink breast and crown, and black primaries and secondaries”



Source: ICML2016, “Generative Adversarial Text to Image Synthesis”

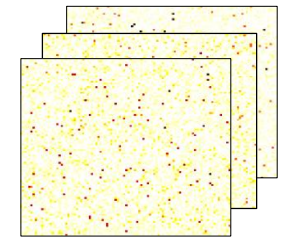
(Conditional) GAN Example

Input: number 2

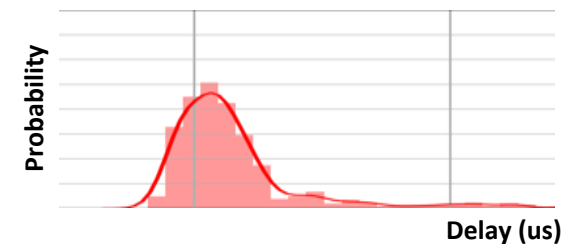


Source: NIPS2014, “Semi-supervised Learning with Deep Generative Models”

(Conditional) VAE Example



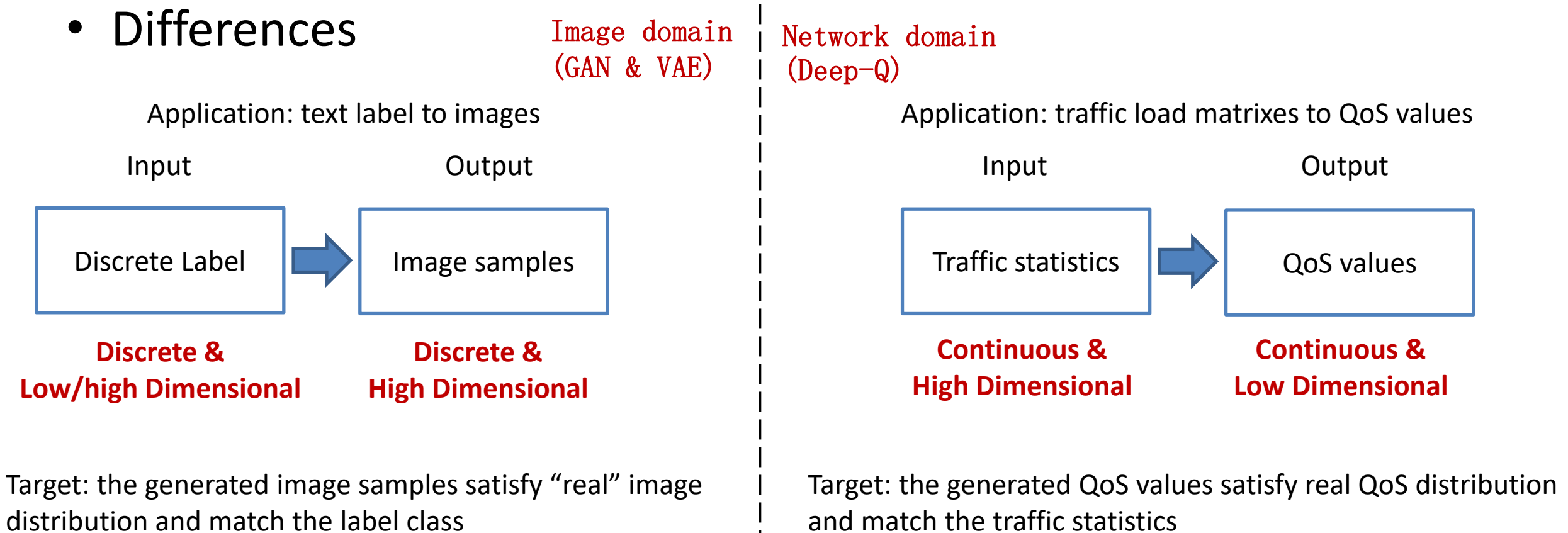
Input: traffic load matrixes



Deep-Q

Key Technology: Deep Generative Network

- Differences

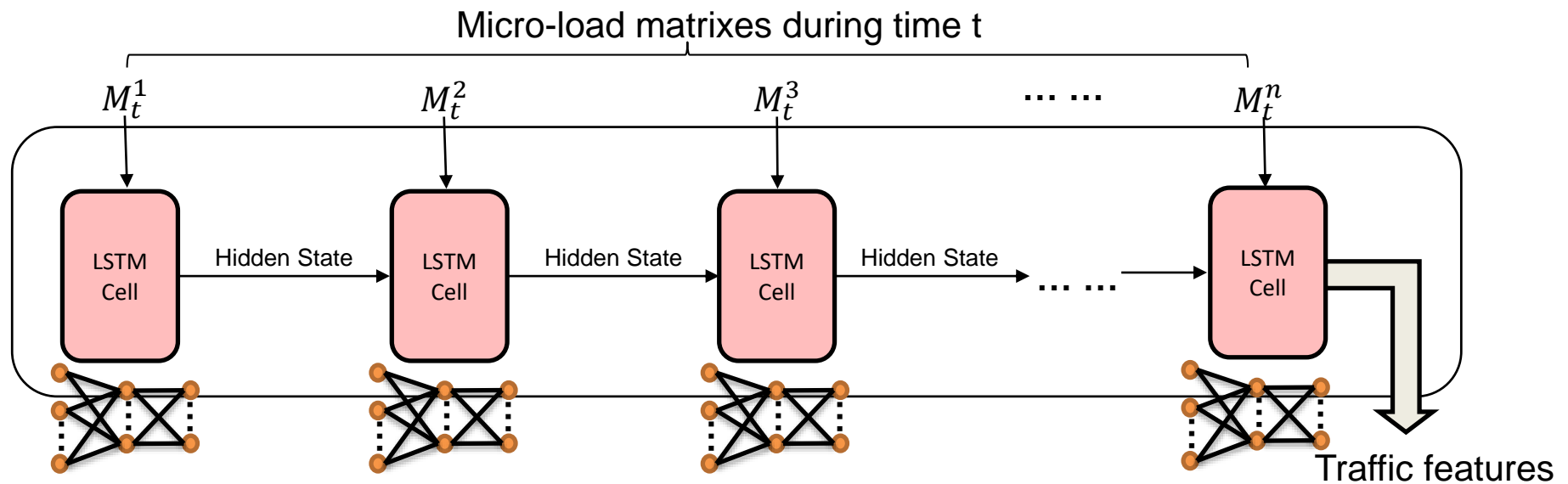


Deep-Q requires a high accuracy on the output distribution, but GAN & VAE do not apply!

Deep-Q Solution

- 1. Handle the continuous high-dimensional input
 - Extract traffic features from a sequence of high-dimensional traffic load matrixes

LSTM (Long Short Term Memory) module: a state-of-the-art deep learning method to learn features from a data sequence



Deep-Q Solution

- 2. Handle the continuous low-dimensional output
 - Challenge: **high accuracy** is required for QoS distribution inference
 - Solution: a new metric “Cinfer loss” to **accurately quantify the QoS distribution error**

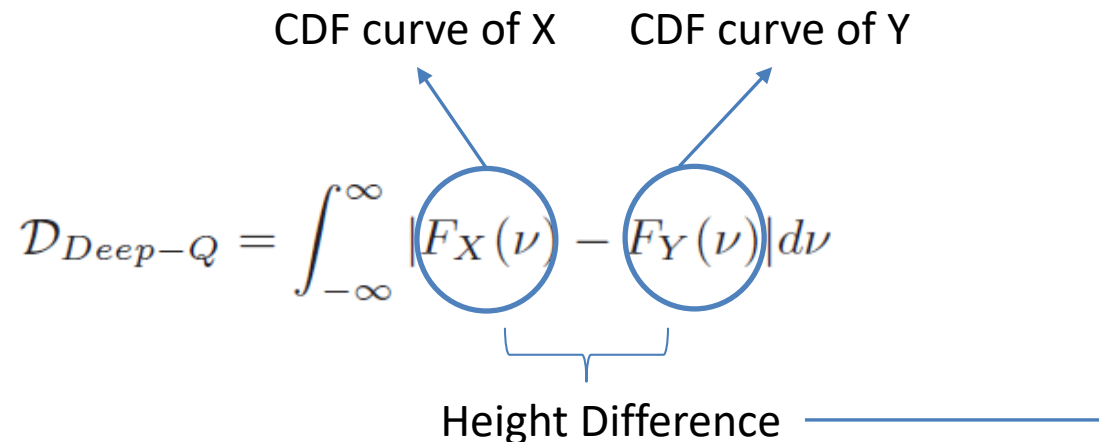
X: Inferred QoS distribution

Y: Target QoS distribution

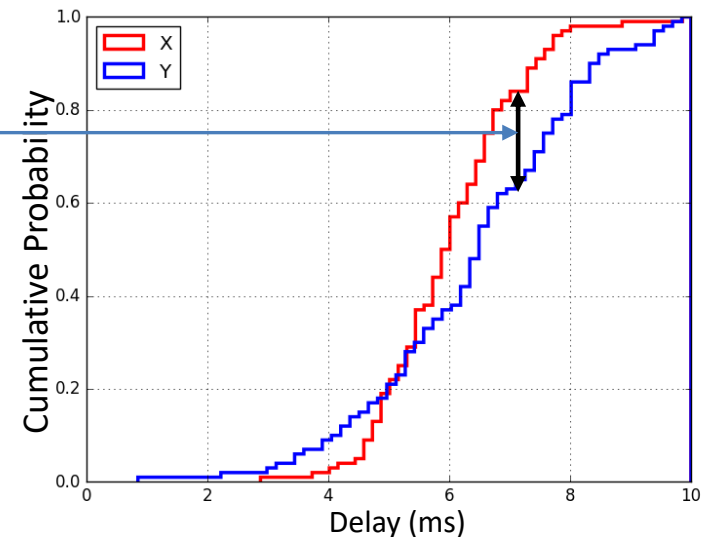
CDF curve of X CDF curve of Y

$$\mathcal{D}_{Deep-Q} = \int_{-\infty}^{\infty} |F_X(\nu) - F_Y(\nu)| d\nu$$

Height Difference



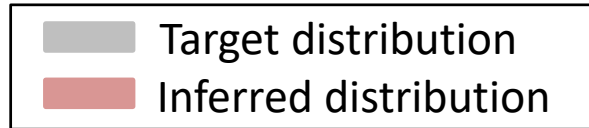
CDF (Cumulative Distribution Function)



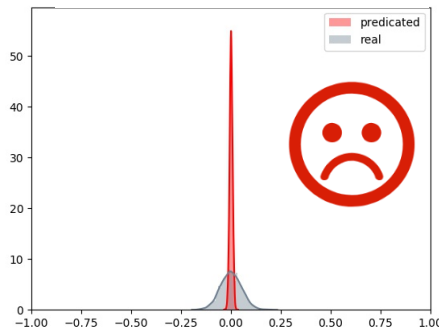
Deep-Q Solution

- Deep-Q: A stable & accurate inference engine
 - Built upon VAE (Stable) and augmented with Cinfer Loss (Accurate)

A simple example of learning ability:



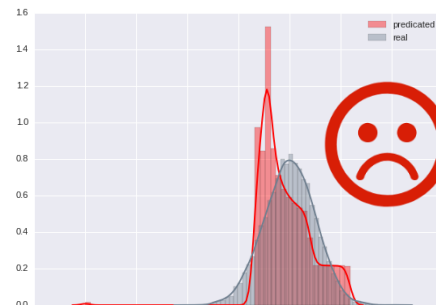
VAE: **Stable** but **Inaccurate**



L2 Loss of VAE

$$D_{VAE} = \frac{1}{n} \sum_{i=1}^n (x_i - y_i)^2$$

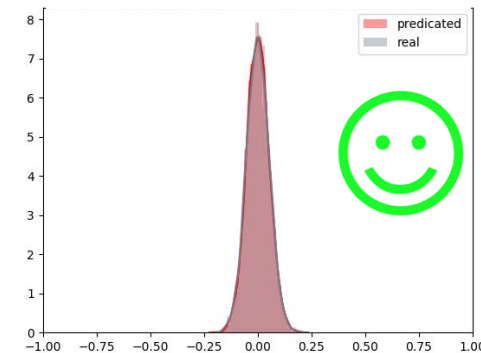
GAN: More **accurate** but **unstable**



KL Loss of GAN

$$D_{GAN} = \mathbb{E}_{x \sim \mathcal{P}(X)} \log \frac{\mathcal{P}(X)}{\mathcal{P}(Y)}$$

Deep-Q: **Stable & Accurate**



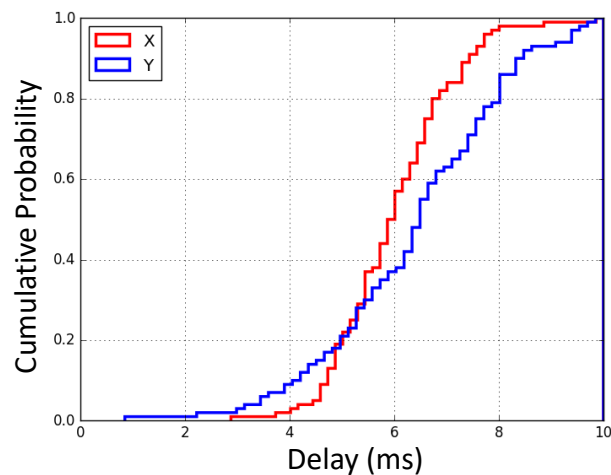
Cinfer Loss of Deep-Q

$$\mathcal{D}_{Deep-Q} = \int_{-\infty}^{\infty} |F_X(\nu) - F_Y(\nu)| d\nu$$

Deep-Q Solution

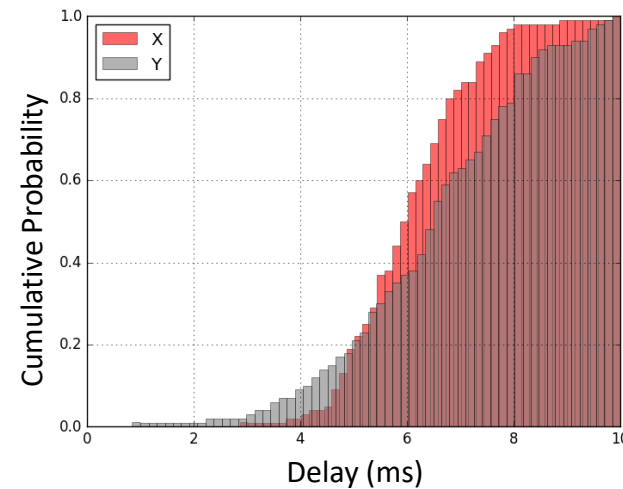
- Cinfer-Loss computation for training
 - The exact computation is NP-hard
 - The approximation must be **fully differentiable** to compute gradients for training
- Step 1: Discretization

From integral to a discrete sum of bins



$$\int_{-\infty}^{\infty} |F_X(\nu) - F_Y(\nu)| d\nu$$

Discretization
→

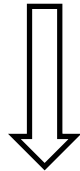


$$\lim_{b \rightarrow \infty} \sum_{k=1}^b |h_X(k) - h_Y(k)|$$

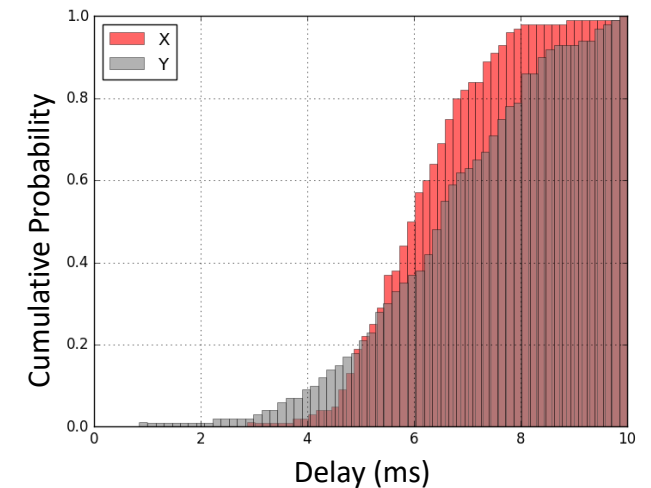
Deep-Q Solution

- Cinfer-Loss computation for training
 - The exact computation is NP-hard
 - The approximation must be **fully differentiable** to compute gradients for training
- Step 2: Bin Height Computation– required to be differentiable
- An intuitive method:
 - Calculate the located bin index of each sample & Count the sample number per bin

$$BinIndex(X_i) = Ceil(\frac{X_i}{BinWidth})$$



Ceil function is non-differentiable & difficult to approximate!



Deep-Q Solution

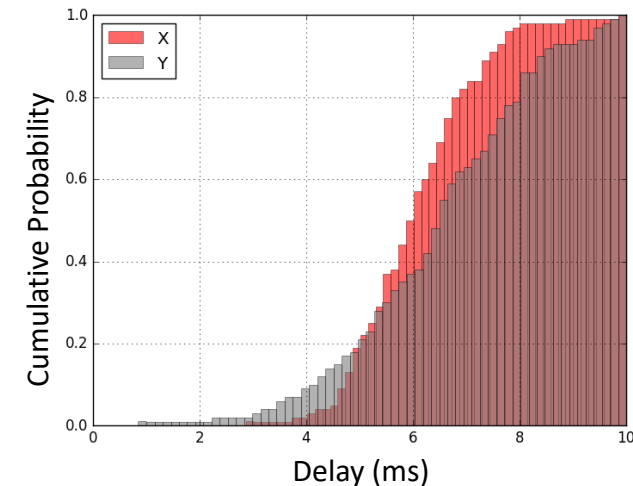
- Cinfer-Loss computation for training
 - The exact computation is NP-hard
 - The approximation must be **fully differentiable** to compute gradients for training
- Step 2: Bin Height Computation– required to be differentiable
- A differentiable method with some math tricks (borrowed from deep learning)

Step 1): Use *Sign* function

$$h_X(k) = (n - \sum_{i=1}^n \text{Sign}(x_i - l_k)) / (2n)$$

Step 2): Approximate *Sign* function with *tanh* 😊

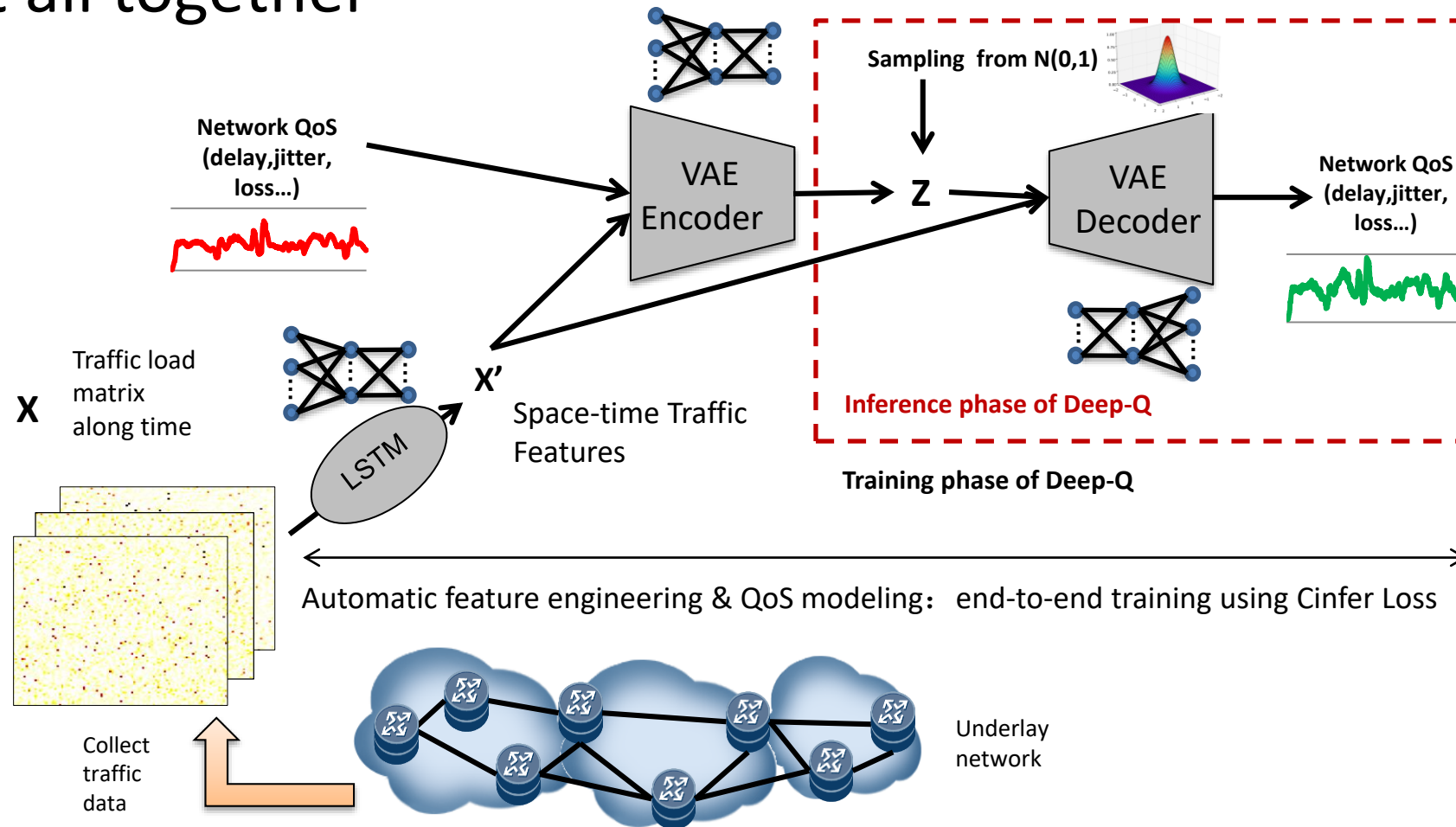
$$h_X(k) = (n - \sum_{i=1}^n \lim_{\beta \rightarrow \infty} \tanh(\beta(x_i - l_k))) / (2n)$$



Approximation error < 10^{-5} in experiments

Deep-Q Solution

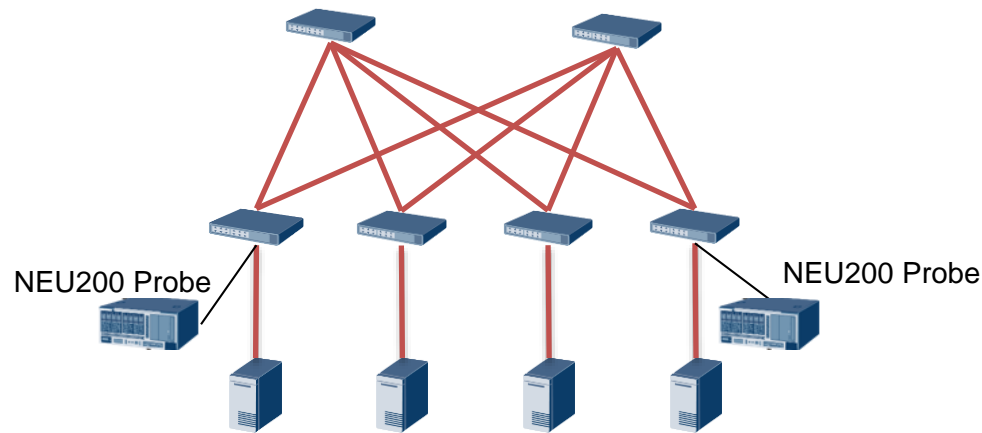
- Put it all together



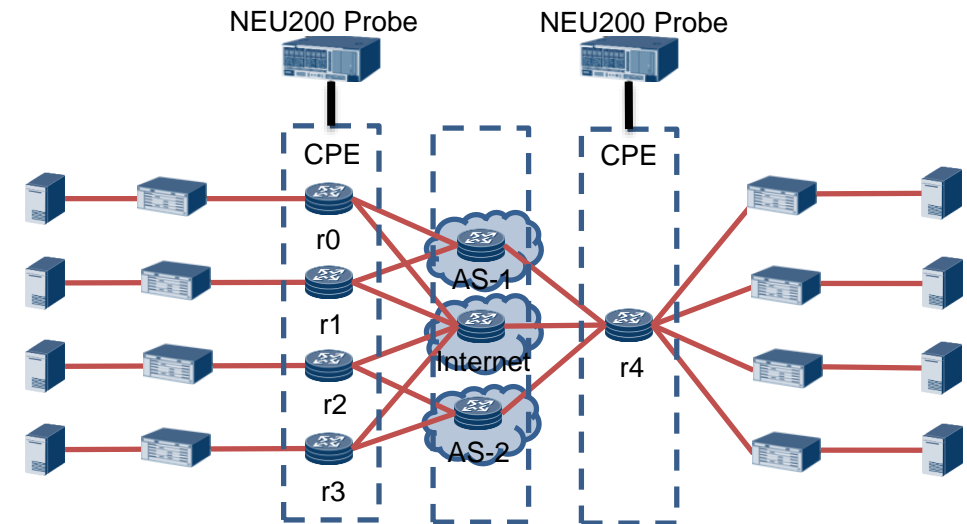
Experiment Setup

- Testbed Topology

Experiment topology of data center network



Experiment topology of overlay IP network

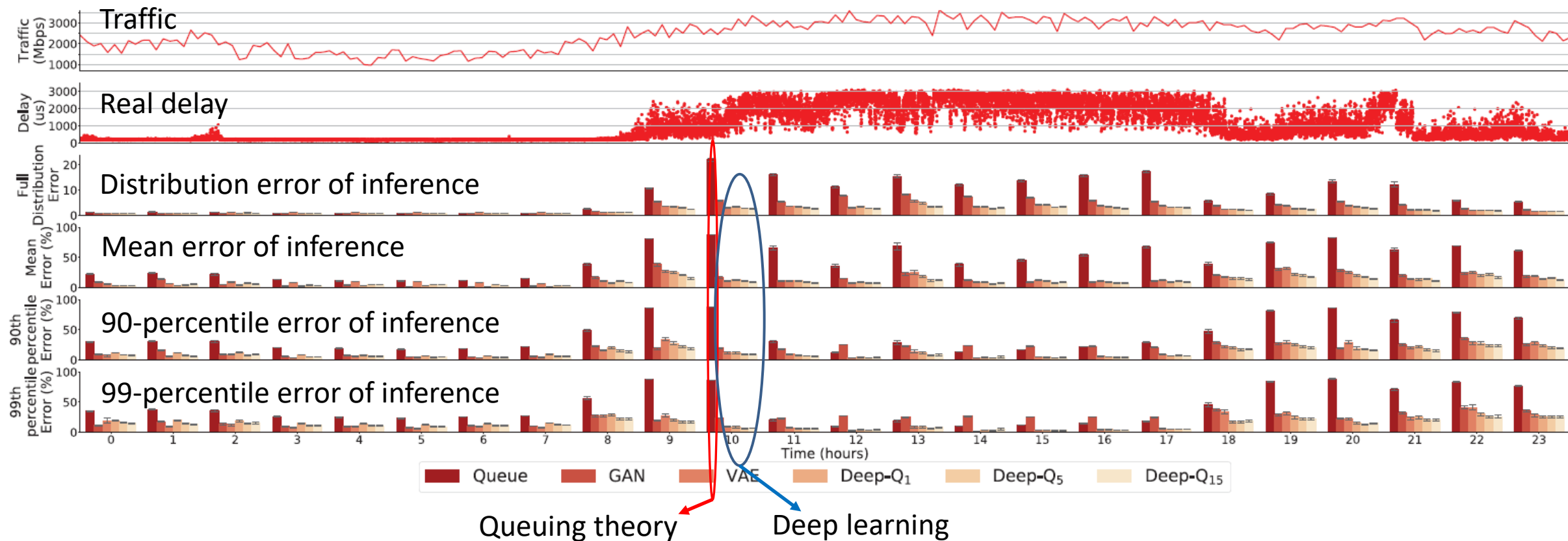


- Traffic traces: WIDE backbone network [1]
 - Training set: 24 hours of traffic traces on April 12, 2017
 - Test set: 24 hours of traffic traces on April 13, 2017
- Neural network: TensorFlow implementation with 2 hidden layers

[1] Traffic traces are public available at <http://mawi.wide.ad.jp/mawi/>

Experiment Results

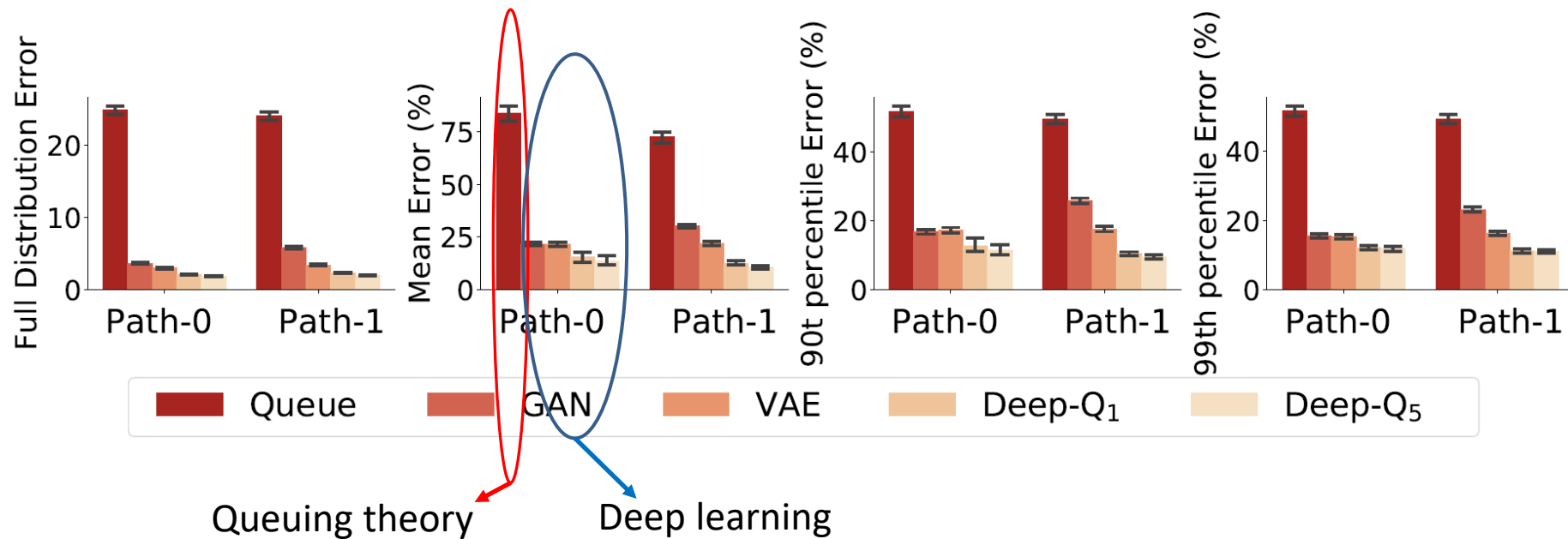
- Delay Inference in Datacenter Topology



1. Deep learning methods achieve **on average 3x higher accuracy** over Queueing theory
2. Deep-Q achieves the **lowest errors and most stable** performance over all cases

Experiment Results

- Packet Loss Inference in Overlay IP Topology



1. Deep learning methods achieve **on average 3x higher accuracy** over Queueing theory
2. Deep-Q achieves the **lowest errors and most stable** performance over all cases

Deep-Q **inference speed < 10ms** for network scale < 200 nodes

Conclusion

- Deep-Q: an **accurate, fast and low-cost** QoS inference engine
 - **Automation**: LSTM module for auto traffic feature extraction
 - **High stability**: an extended VAE inference structure with the encoder and decoder
 - **High accuracy**: a new metric “Cinfer loss” to accurately quantify the QoS distribution error
- Future vision:
 - Learn device-level QoS models (routers/switches) → scalable network-level QoS models
 - Learn high-level application QoE from traffic traces

Thank You !