IEEE SERVICES 2022

DisCOV: Distributed COVID-19 Detection on X-Ray Images with Edge-Cloud Collaboration

Xiaolong Xu[*], Hao Tian[*], Xuyun Zhang[#], Lianyong Qi[†], Qiang He[‡], Wanchun Dou[§]

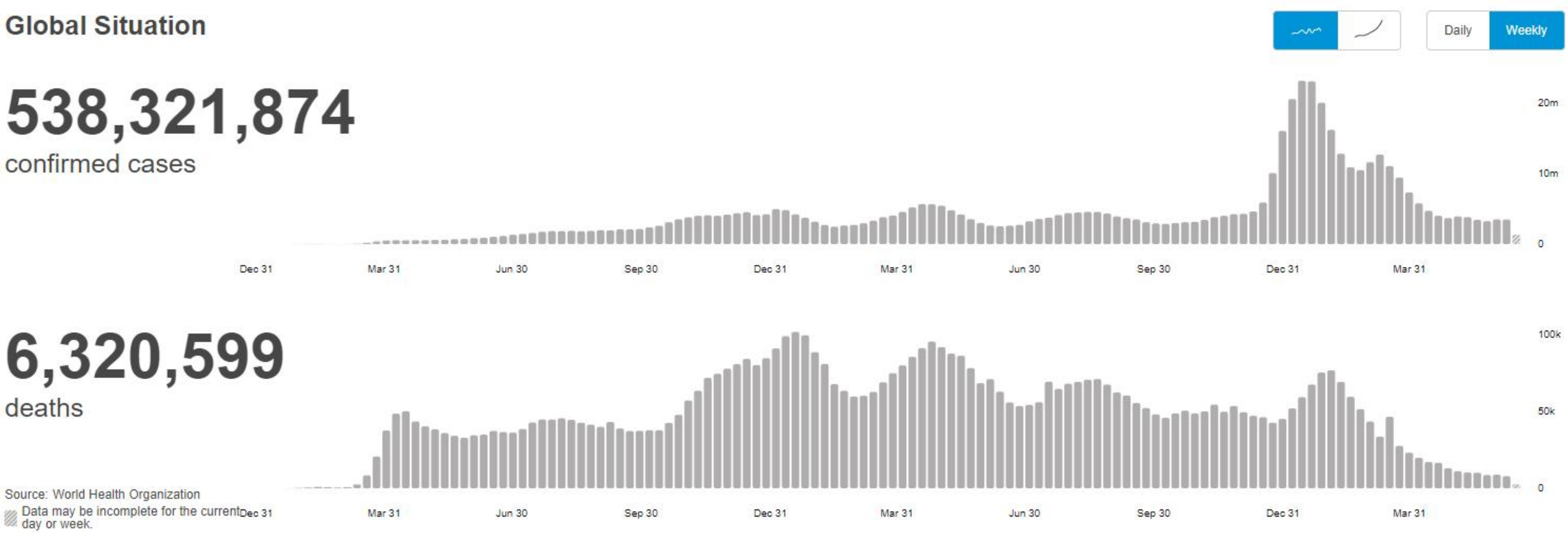
[*]Nanjing University of Information Science and Technology, [#]Macquarie University, [†]Qufu Normal University, [‡]Swinburne University of Technology, [§]Nanjing University

Outline

- **Background**
  - The Outbreak of COVID-19
  - AI-Empowered COVID-19 Detection
- **Motivation**
- **System Model and Problem Formulation**
  - Distributed Edge Learning Model and Cloud Aggregation Model
  - System Latency and Energy Consumption Maximization
- **Design of DisCOV**
  - Lightweight Model–Based Distributed Training Algorithm
  - Dynamic Resource Allocation Algorithm
- **Experimental Evaluation**
- **Summary & Conclusions**

The Outbreak of COVID-19

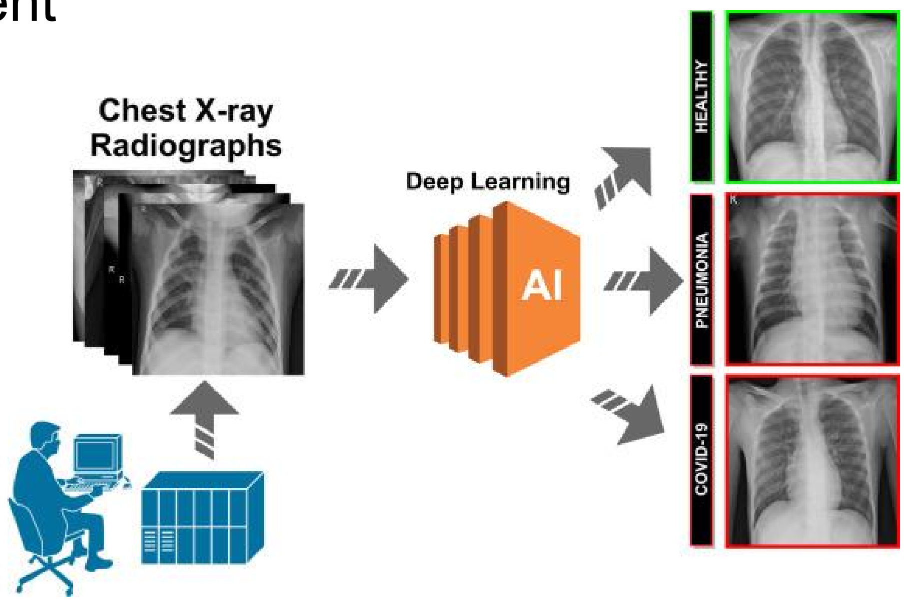
- **COVID-19 takes a devastating impact on**
  - Society
  - Economy
  - Public healthcare



Source: https://covid19.who.int

AI-Empowered COVID-19 Detection

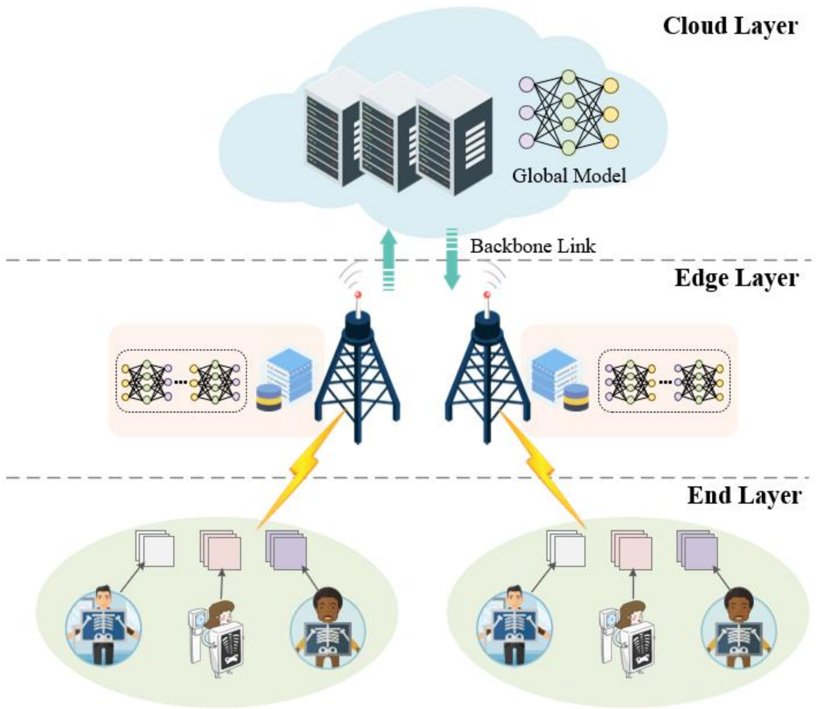
- **Deep learning increases medical image processing capabilities**
  - CNN
- **Chest X-ray (CXR) is capable of executing COVID-19 detection**
  - Rapid
  - Convenient



Motivation

■ Cloud-based vs. edge computing

- Unpredictable remote server and communication latency
- Unbearable bandwidth pressure with massive raw data uploading
- Computational resources near the end devices



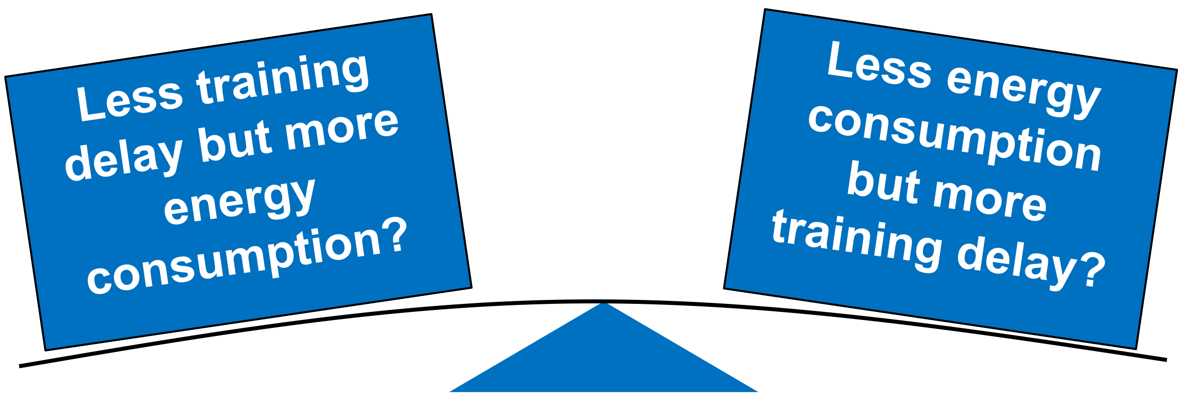
Motivation

■ Training efficiency and resource utilization

- The training process of the DL model is resource-intensive
- The model parameters and required computational resources are generally large
- Edge nodes with limited resources (e.g., processor, memory, and bandwidth) hardly undertake multiple training tasks

Motivation

- **Trade-off between training delay and energy cost**
  - Each edge node may perform one or more training tasks in parallel
  - The inappropriate resource allocation strategies result in longer training delay and higher energy of some tasks



System Model and Problem Formulation

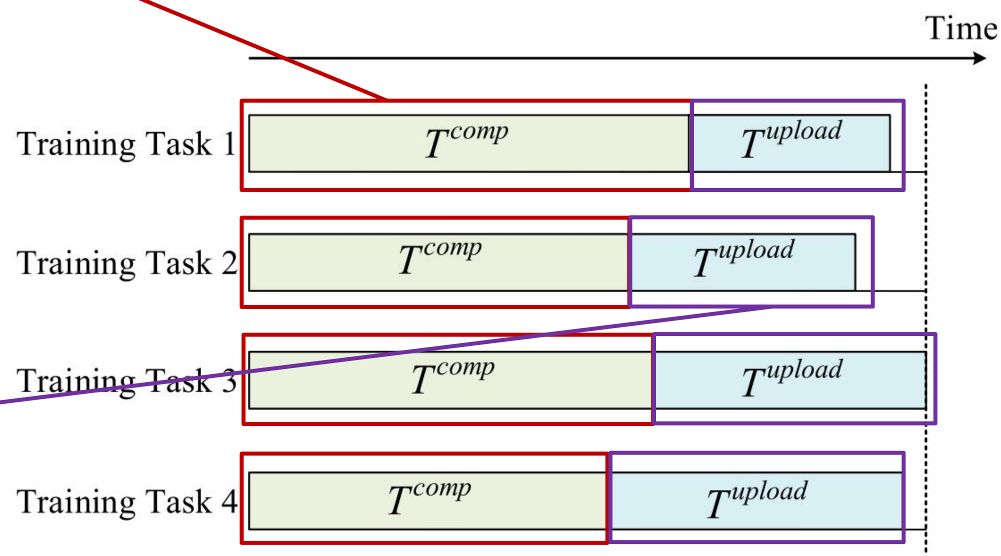
- Distributed edge learning model
- Cloud aggregation model
- Optimization problem formulation

Distributed Edge Learning Model

- **Local training on edge node**

$$T_{m,n}^{comp} = x_{m,n} Q_m L_n \left| D_{m,n} \right| \frac{\xi_m}{\boxed{f_{m,n}}}$$

immediate computing
frequency is the key factor



- **Model parameters uploading for edge node**

$$T_{m,n}^{upload} = x_{m,n} Q_m \frac{c_m}{\boxed{b_{m,n}} \log_2 (1 + r)}$$

immediate bandwidth allocation
is also important

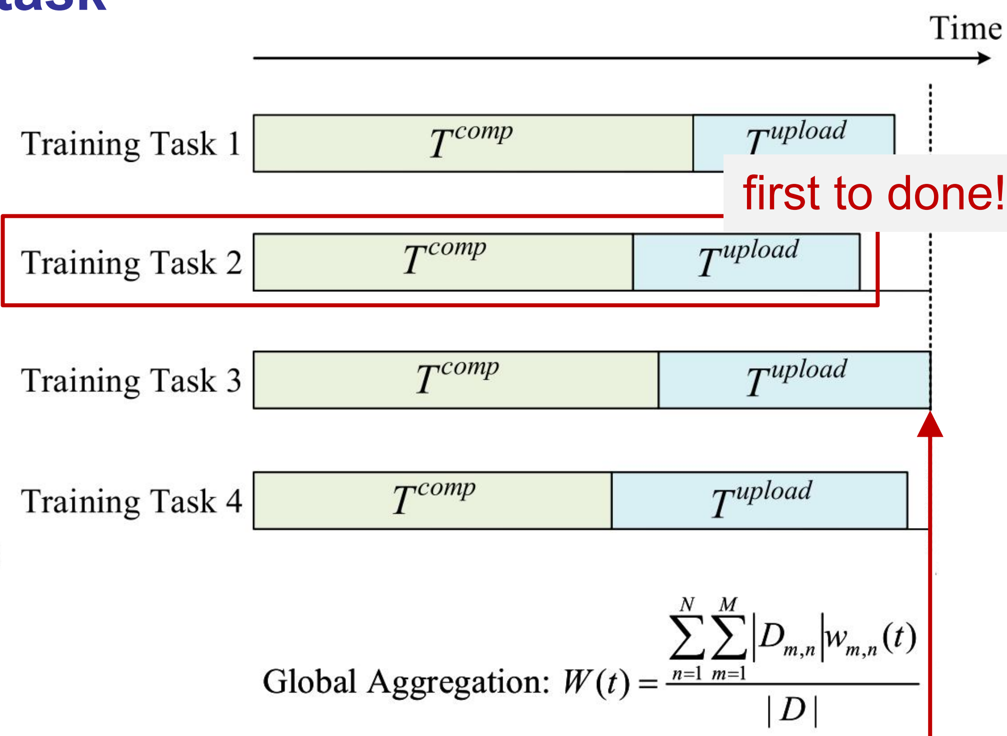
Global Aggregation: $W(t) = \dfrac{\sum_{n=1}^{N} \sum_{m=1}^{M} \left| D_{m,n} \right| w_{m,n}(t)}{|D|}$

Cloud Aggregation Model

- **Cloud aggregation depends on the last finished training task**

$$T_{total}^{e} = G \cdot \underline{\max_{m \in \mathcal{M}, n \in \mathcal{N}} \left\{ T_{m,n}^{comp} + T_{m,n}^{upload} \right\}}$$

$$E_{total}^{e} = G \cdot \sum_{n=1}^{N} \sum_{m=1}^{M} p_n^c T_{m,n}^{comp} + p_n^u T_{m,n}^{upload}$$



But aggregation conducts when task 3 finishes

Optimization Problem Formulation

- **Goal: to jointly minimize the training time and energy consumption during the training phase**

$$\min_{(f,b)} \quad \eta_T T_{total} + \eta_E E_{total} \tag{1}$$

s.t.

$$Q_m \in \{0,1\}, \forall m \in \mathcal{M}, \tag{2}$$

$$x_{m,n} \in \{0,1\}, \forall m \in \mathcal{M}, n \in \mathcal{N}, \tag{3}$$

$$0 \leq \sum_{m=1}^{M} x_{m,n} Q_m f_{m,n} \leq F_n, \forall n \in \mathcal{N}, \tag{4}$$

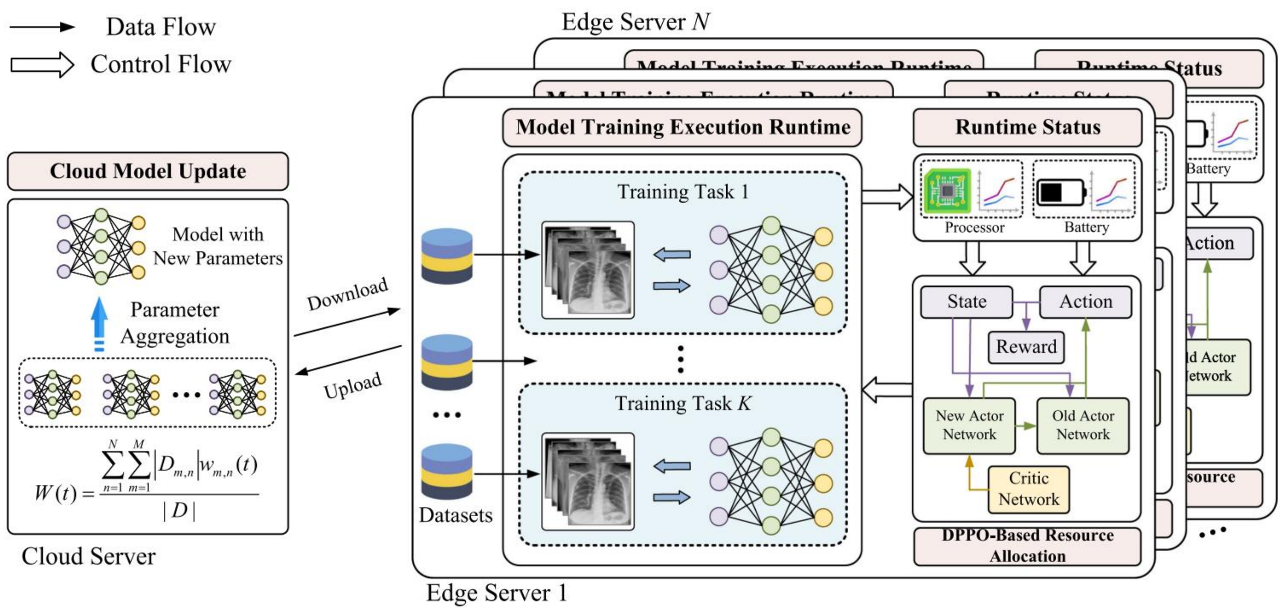
$$0 \leq \sum_{m=1}^{M} x_{m,n} Q_m b_{m,n} \leq B_n, \forall n \in \mathcal{N}, \tag{5}$$

Design of DisCOV

- DisCOV Overview
- Lightweight Model–Based Distributed Training Algorithm
- Dynamic Resource Allocation Algorithm

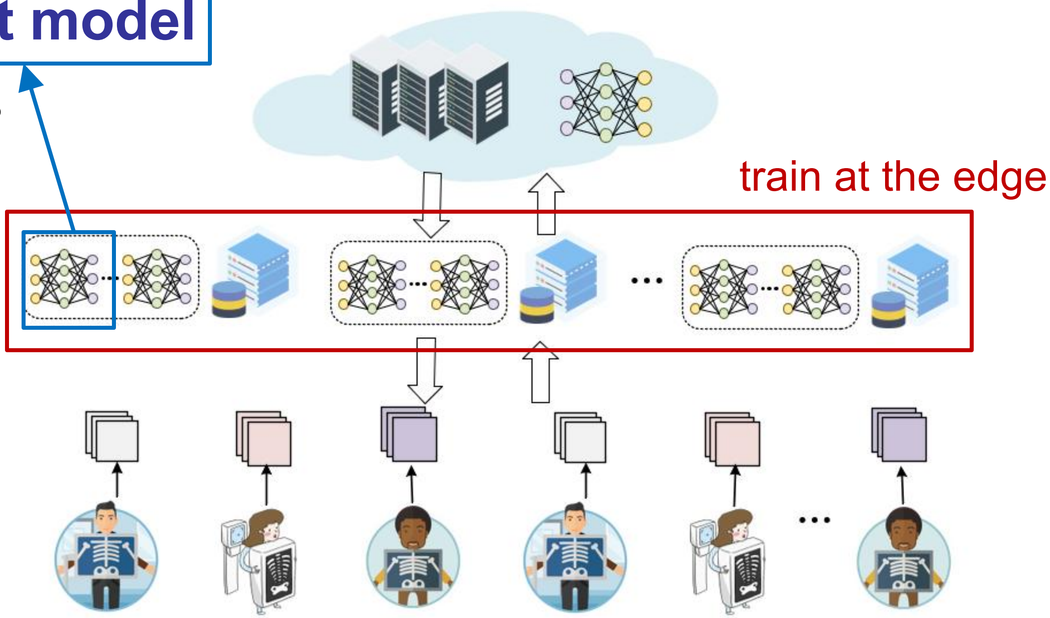
DisCOV Overview

- **Lightweight Model–Based Distributed Training (LDT)**
  - Training in parallel
  - Collaborative training
- **Dynamic Resource Allocation (DRA)**
  - Time-varying resource allocation



Lightweight Model–Based Distributed Training Algorithm

- **Model training conducts at the edge with edge-cloud collaboration**
  - end device with constrained computation and storage
  - raw data uploading exhausts bandwidth resources
  - split total data transmit to each edge node to release computing pressure
- **Training with lightweight model**
  - less model parameters
  - less computations



Lightweight Model–Based Distributed Training Algorithm

- **We propose the LDT to perform the training phase**

training task in edge nodes processes data samples in parallel

update the model parameter

Algorithm 1. LDT

Input: $\mathcal{M}, \mathcal{N}, G, L_n, D_{m,n}, W, w_{m,n}, Loss_{m,n}(w)$;
Output: Global parameter W;
`// Initialization`
Initialize G and L_n;
Initialize W, $w_{m,n}$, and $Loss_{m,n}(w)$;
for *iteration* = 1 *to* G **do**
  for *each training task in parallel* **do**
    `// Edge training`
    for $l = 1$ *to* L_n **do**
      for $k \in D_{m,n}$ **do**
        ES n chooses one local sample $k \in D_{m,n}$ from CXR device m;
        Calculate the loss function $loss_k(w)$ of one sample k;
      end
      Calculate the loss function by (5);
      Update the local parameters $w_{m,n}$ by (6);
    end
    Transmit the local parameters $w_{m,n}$ from ES n to the cloud;
  end
  `// Cloud aggregation`
  Update the glocal paramaters of the aggregation by (13);
end
return W;

Dynamic Resource Allocation Algorithm

■ The formulated problem can naturally be expressed as Markov decision processes (MDP)

- States

$$S(t) = \{C(t), F(t), K(t), R(t), B(t)\}$$

  - training data size from CXR devices
  - required computing resources
  - size of model parameters
  - available resources of ESs

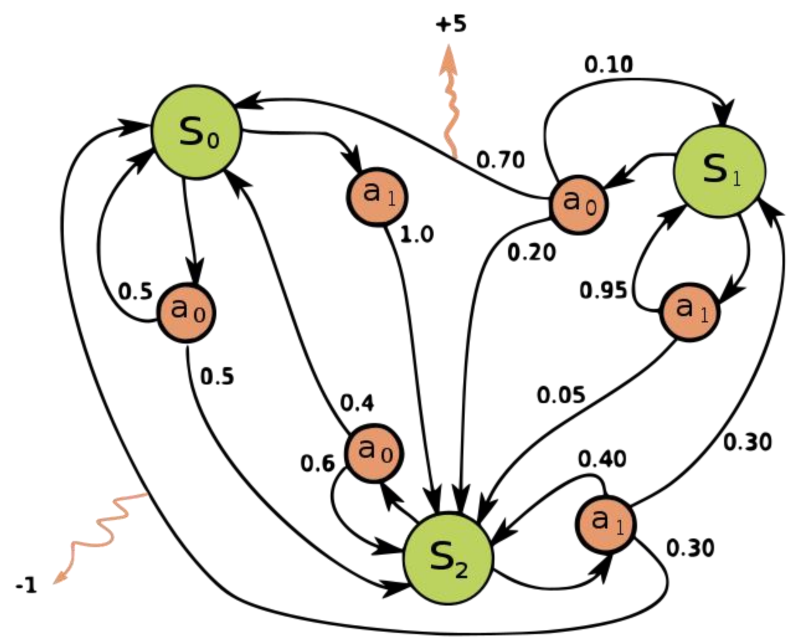
- Actions

$$A(t) = \{f(t), b(t)\}$$

  - allocated computation resources
  - allocated bandwidth resources

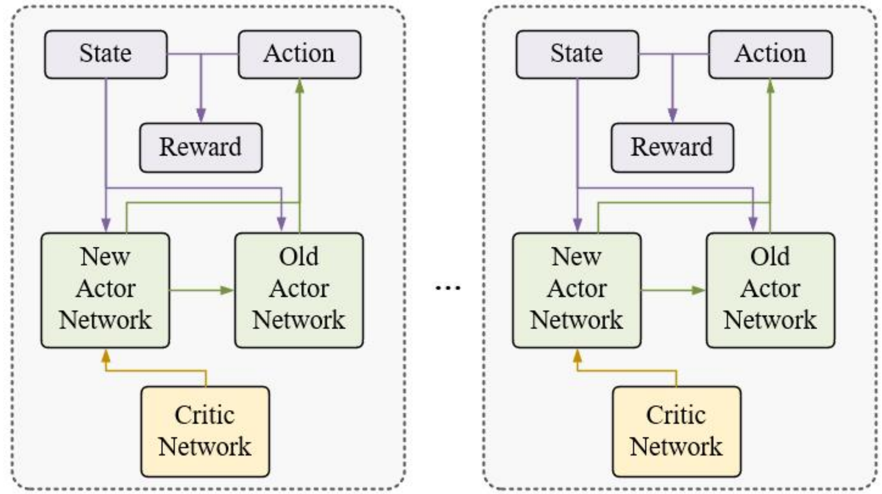
- Reward

$$R(S(t), A(t)) = -(\eta_T T + \eta_E E)$$

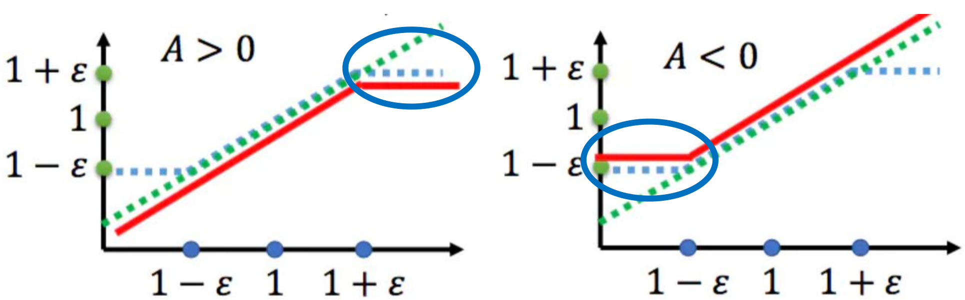


Dynamic Resource Allocation Algorithm

- **DRL-based resource allocation algorithm to dynamically dispatch computing and bandwidth**
  - environment-aware
  - actions perform at each time slot



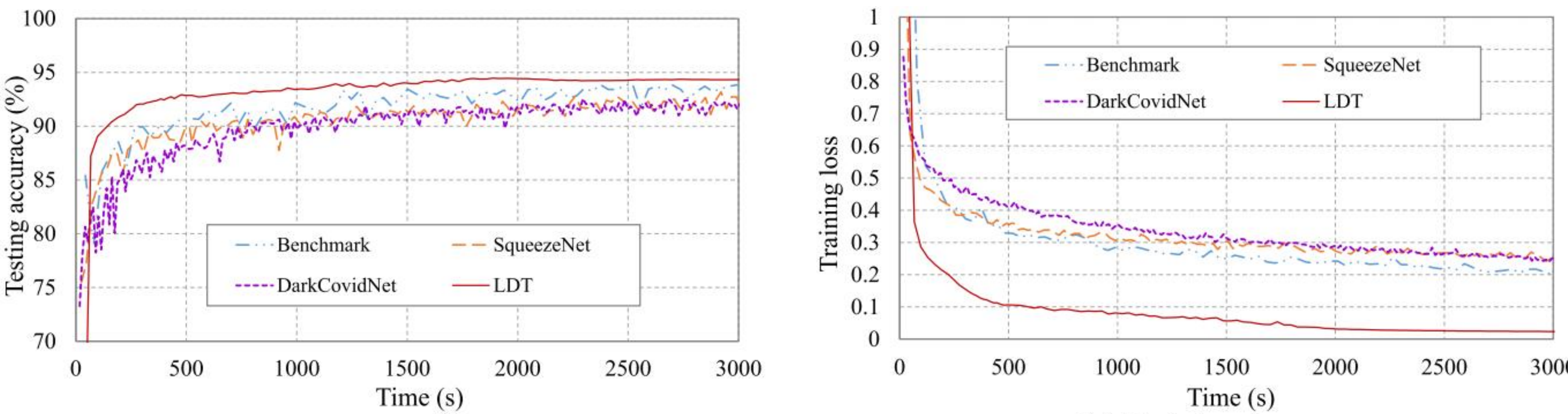
$$L_C(\theta) = E[\min(\varrho(\theta)A_{\theta'}, \boxed{clip(\varrho(\theta), 1-\delta, 1+\delta)A_{\theta'})})]$$



avoid to over-update
improve the performance stability

Experimental Evaluation

■ Training performance on different methods



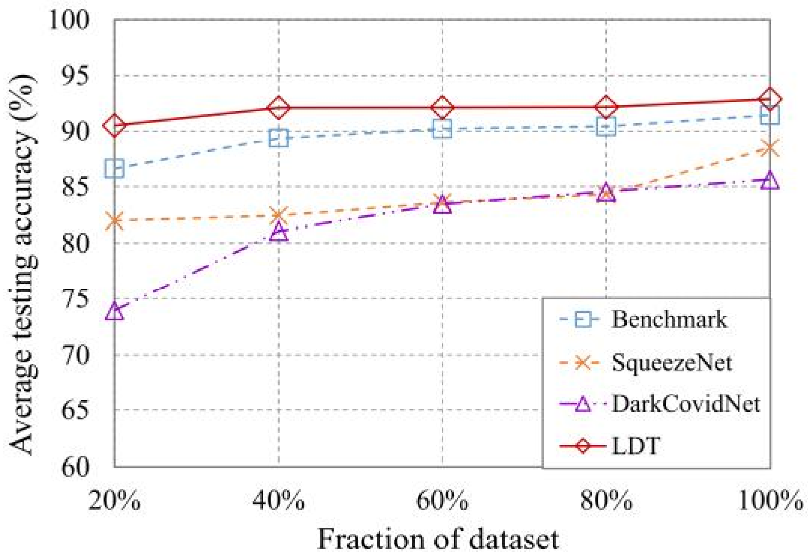
(a) Testing accuracy.

(b) Training loss.

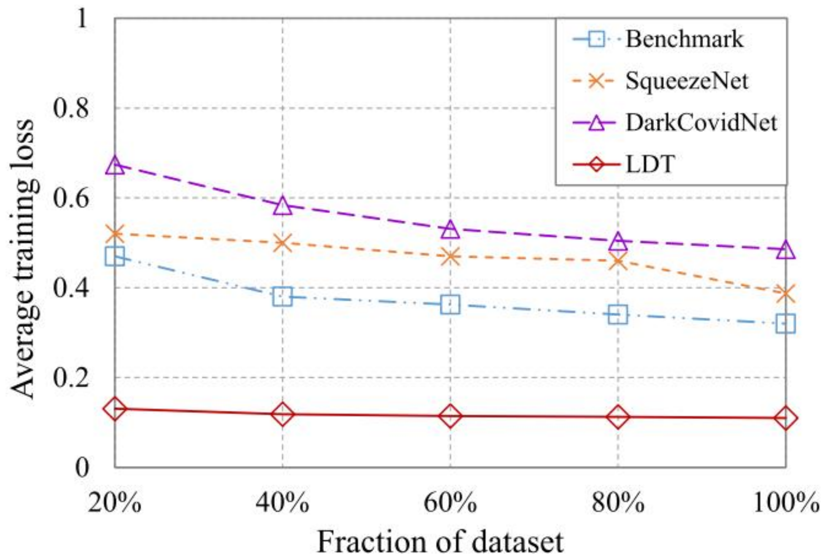
Expected testing accuracy	85%		90%		93%		94%	
	Elapsed time	Iterations	Elapsed time	Iterations	Elapsed time	Iterations	Elapsed time	Iterations
Benchmark	118.45s	3	447.44s	12	1210.84s	33	-	-
SqueezeNet	116.97s	5	566.49s	26	2921.37s	126	-	-
DarkCovidNet	200.08s	16	785.56s	60	-	-	-	-
LDT	**66.98s**	**2**	**160.18s**	**5**	**679.5s**	**30**	**1392.62s**	**53**

Experimental Evaluation

■ Training performance on different fractions of dataset



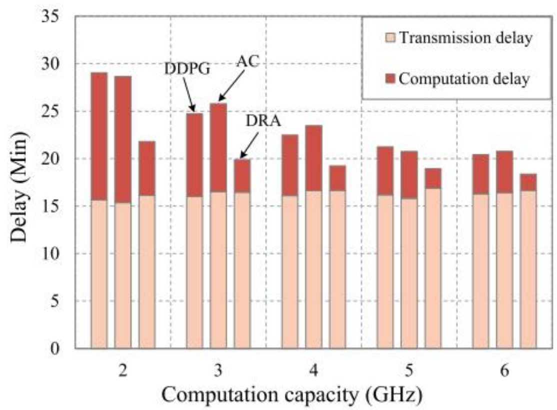
(a) Average testing accuracy.

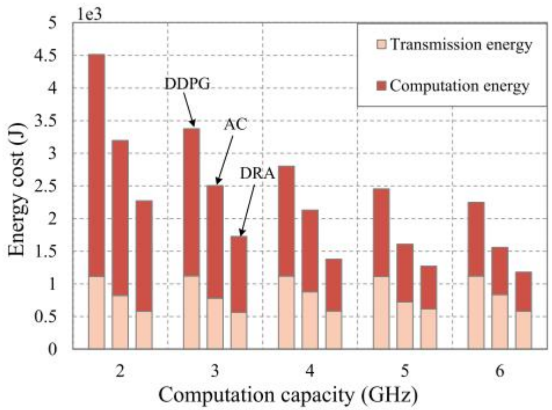


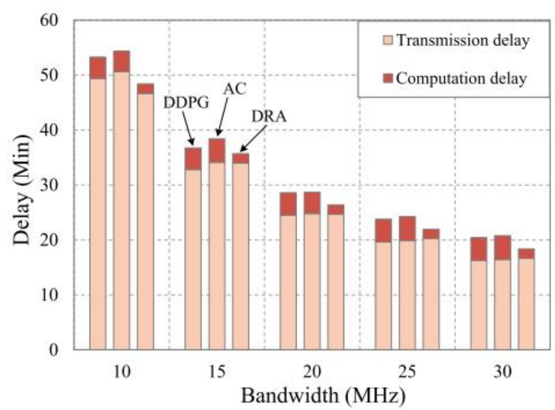
(b) Average training loss.

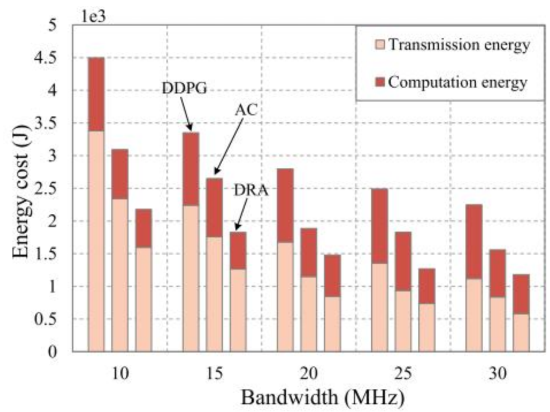
Experimental Evaluation

- **Performance of training delay and energy cost on different computation and bandwidth resources**









Summary & Conclusions

- **DisCOV**
  - LDT
    - Lightweight model-based training with edge-cloud collaboration
    - Training in parallel with cooperation of edge nodes
  - DRA
    - Original problem models into MDP problem
    - Dynamic allocation of computing and communication resources
  - Faster training speed with *64%* reduction of data transmission
- **Future work**
  - Distributed training architecture with decentralized mode
  - Fine-grained training task scheduling scheme
  - Implementation of the prototype

Thanks!