Target-based Resource Allocation for Deep Learning Applications in a Multi-tenancy System

Wenjia Zheng[1], Yun Song[1], Zihao Guo[1], Yongchen Cui[1], Suwen Gu[1], Ying Mao[1], Long Cheng[2]

Fordham University[1] - University College Dublin[2]

Outline

- Introduction & Background
- Motivation
- *TRADL* Modules
- System Evaluation
- Conclusion

Deep Learning is Reshaping Our World

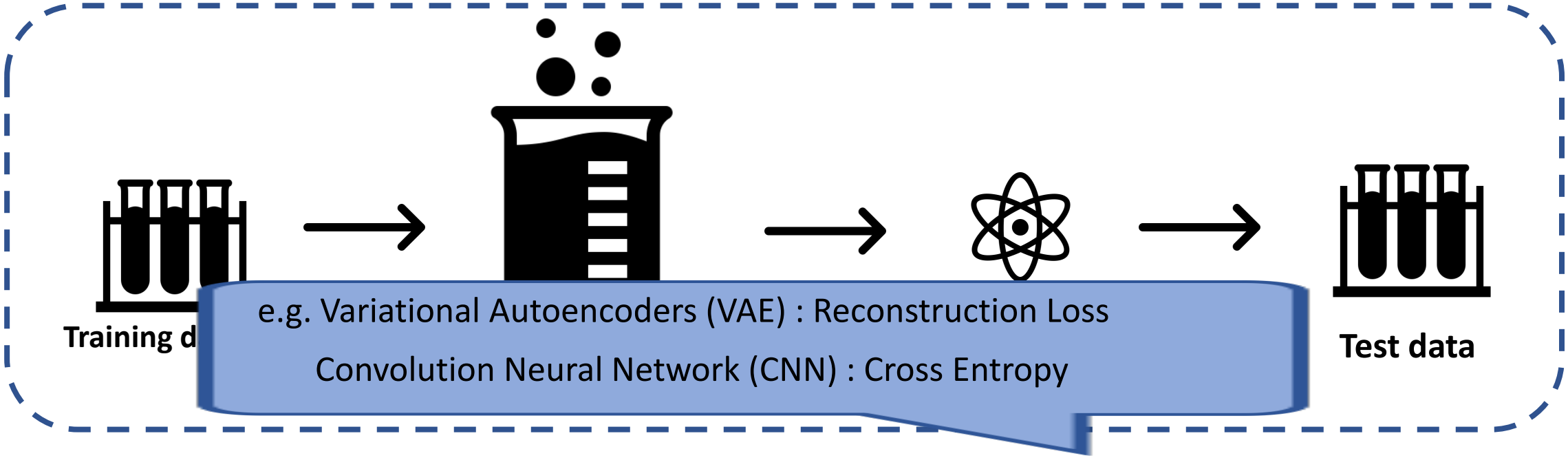
Self-Driving Cars

Face Recognition

Voice Controlled Assistance

Automatic Language Translation

Training process of Deep Learning

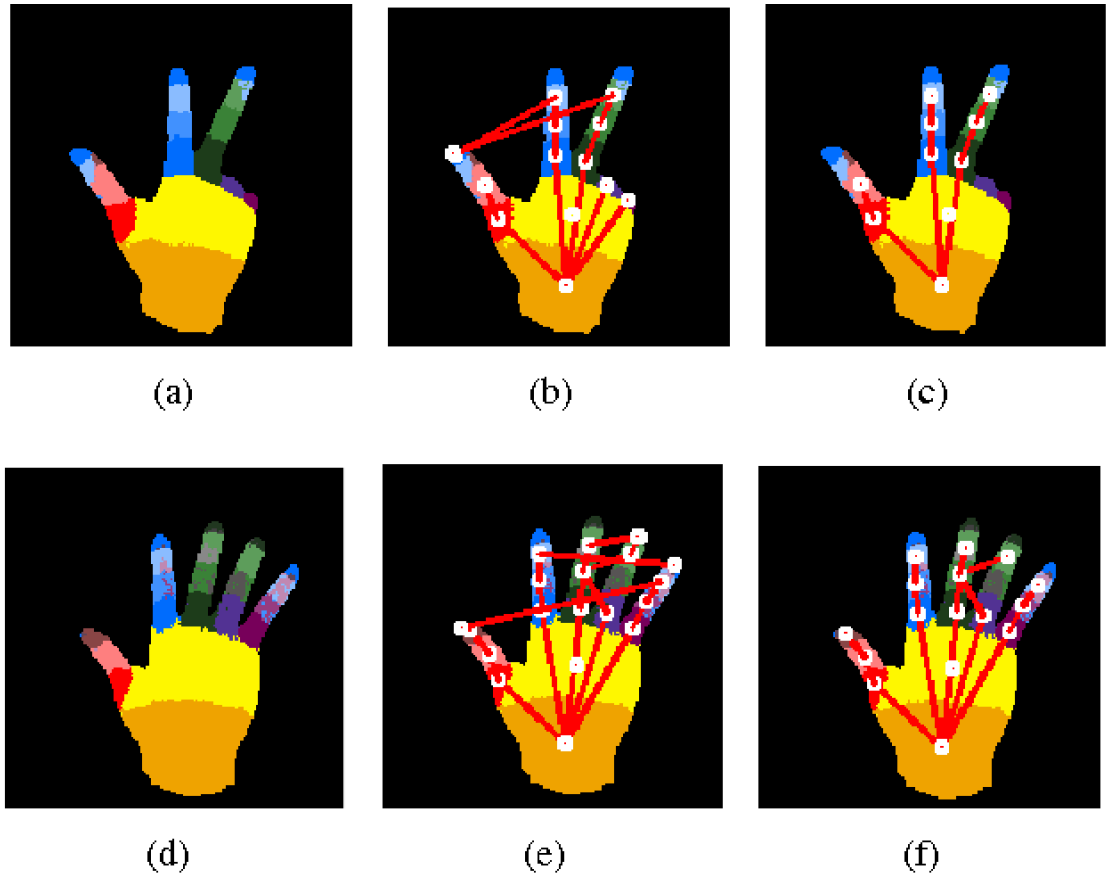


Evaluation: Loss function (predefined)

Target: Global minimum

Error-tolerated DL application

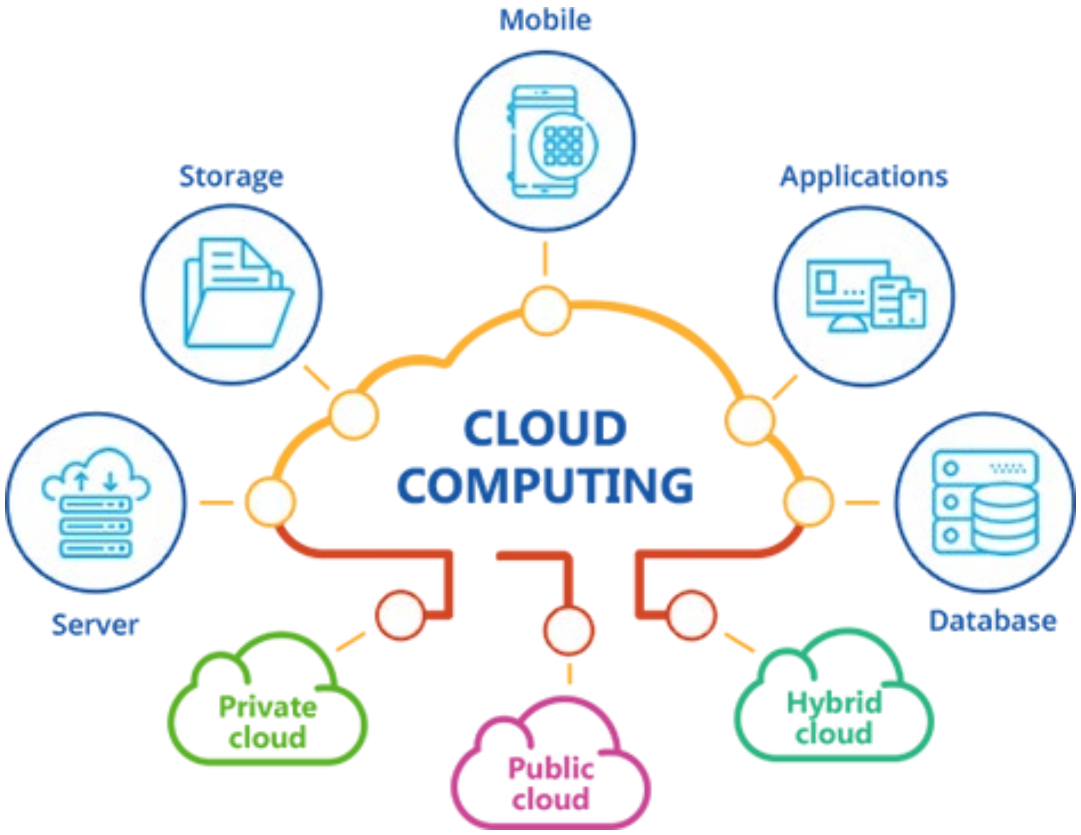
Hand pose estimation



t need to achieve 100% accuracy

sumption:
ers can tolerate a certain
el of error rate

Limited Resources on the Cloud

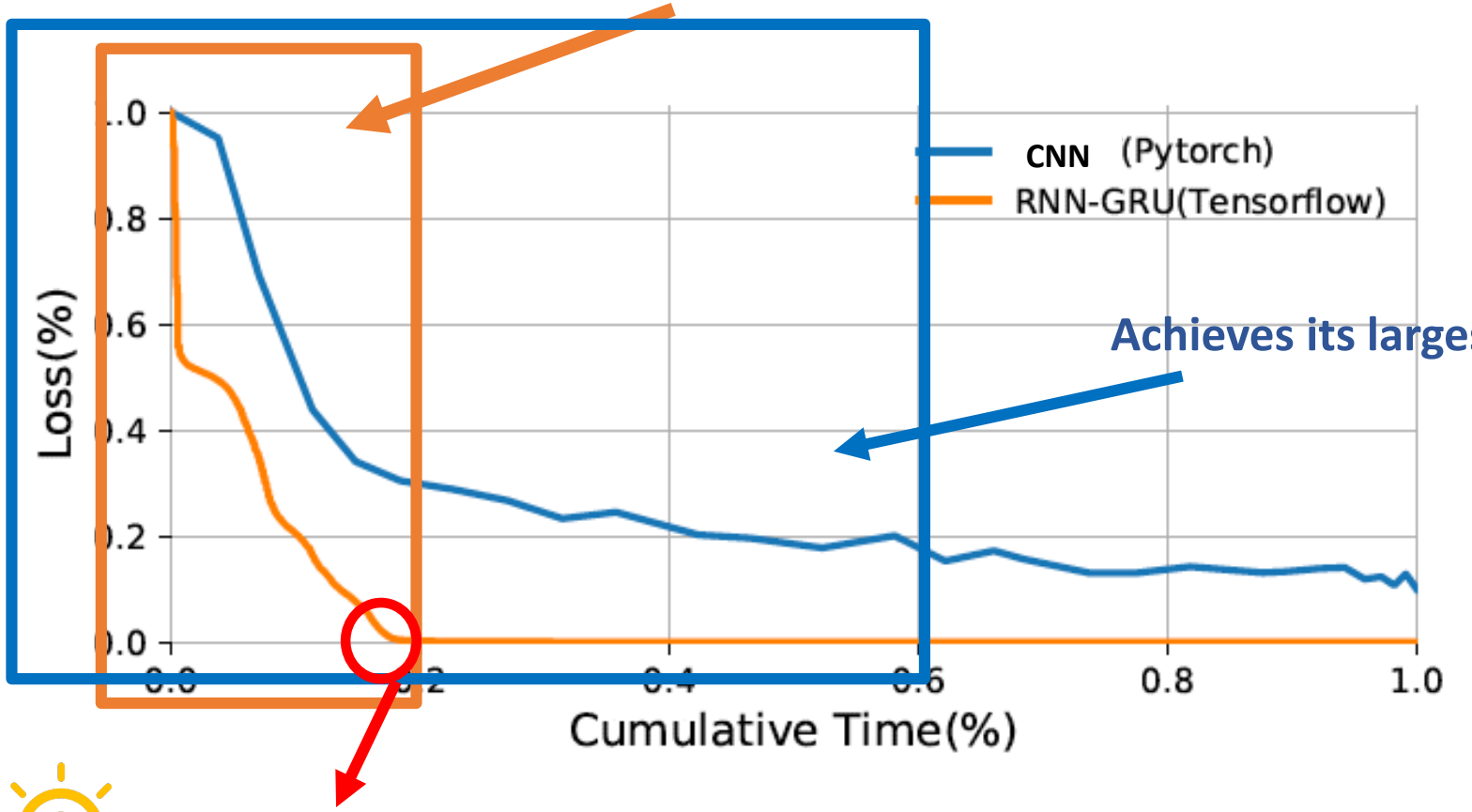


Outline

- Introduction & Background
- Motivation
- *TRADL* Modules
- System Evaluation
- Conclusion

Motivation

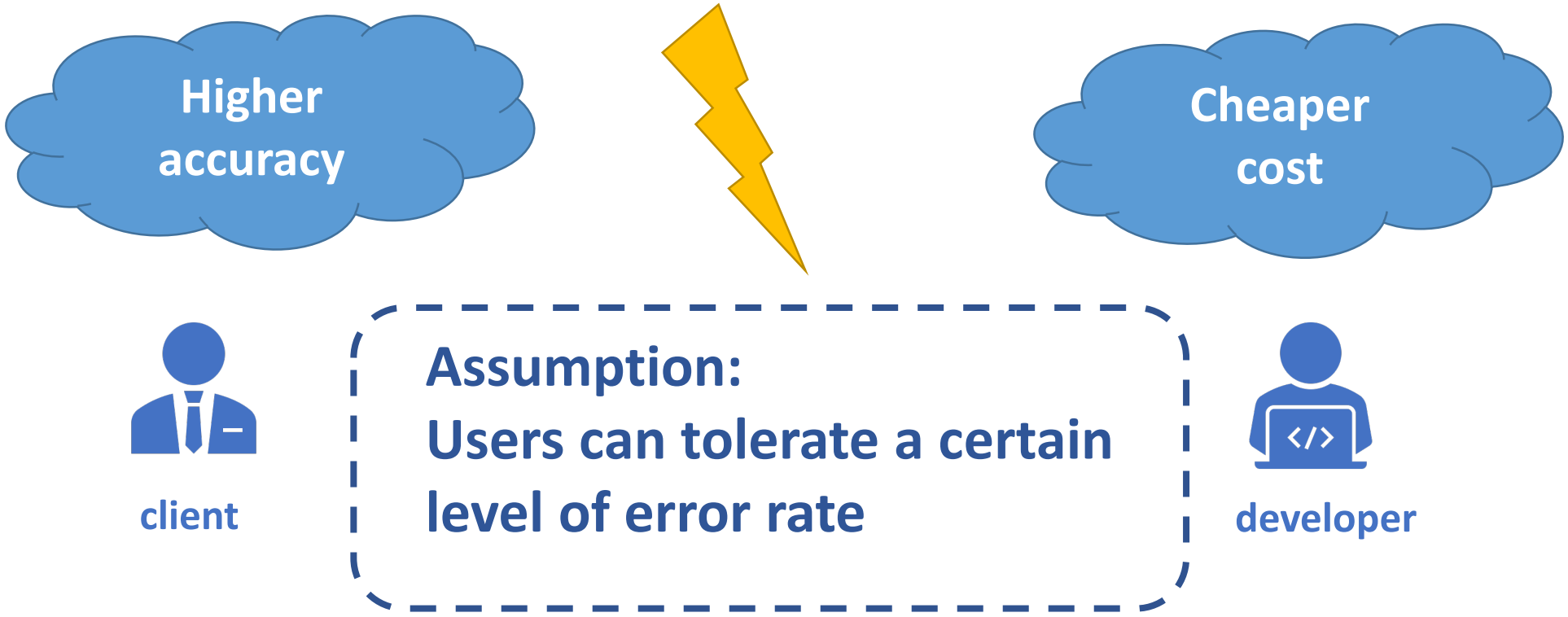
Achieves its largest reduction at 20% of the time



Achieves its largest reduction at 60% of the time

Allocate less resource & shift resource to other learning tasks

Trade-off relationship

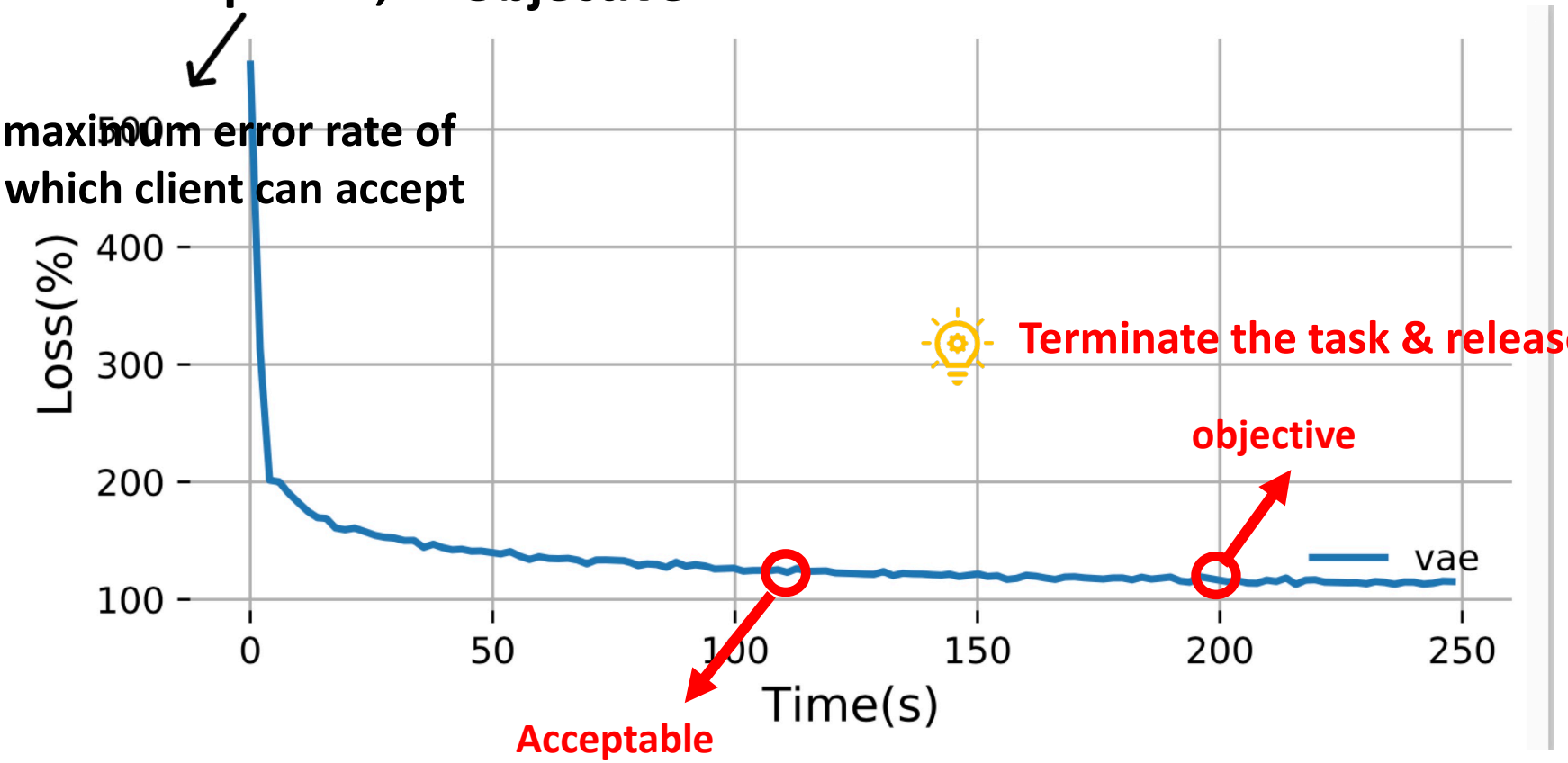


Two-tier target

Two-tier target

<Acceptable, Objective> → **The value of loss which developer satisfy**

The maximum error rate of loss which client can accept



Terminate the task & release resource

objective

Acceptable

Allocate less resource & shift resource to other learning tasks

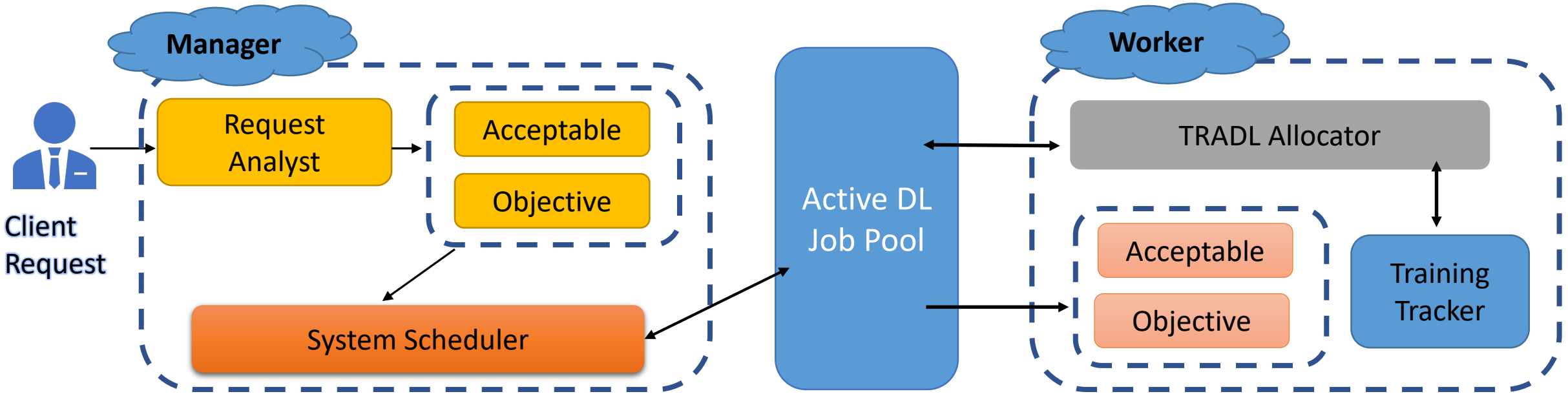
Outline

- Introduction & Background
- Motivation
- *TRADL* Modules
- System Evaluation
- Conclusion

TRADL Modules

「Interact with clients and assign tasks to the workers」

「Execute the tasks and generate the results」



「Control the resource allocation globally & label the training job with Acceptable and Objective」

Resource Assignment Algorithm

One job

1. Resources consumed by other processes → **Using resource < α * Total Resource** → **Free compete**

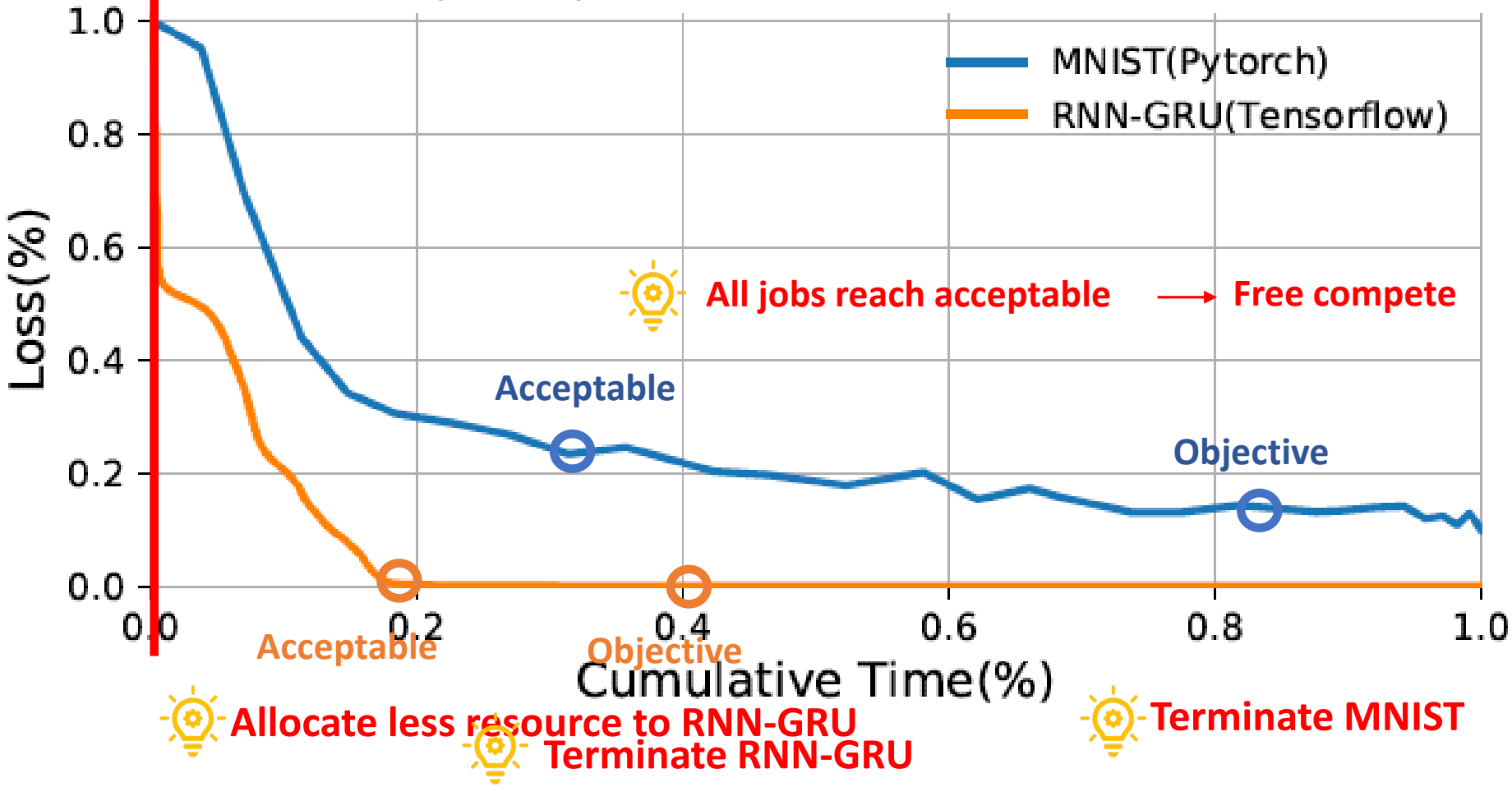
Allocate less resource & shift resource to other learning tasks

Terminate the task & release resource

Resource Assignment Algorithm

Multi jobs

1. Resources consumed by other processes → **Sum of using resource < α * Total Resource** → **Free compete**



Outline

- Introduction & Background
- Motivation
- *TRADL* Modules
- System Evaluation
- Conclusion

System Evaluation

- **Workload**

Model	Loss Function	Two-tier Target
Variational Autoencoders (VAE)	Recon. Loss	120, 110
Convolution Neural Network (CNN)	Cross Entropy	0.3, 0.25
Gated Recurrent Unit (GRU)	Quadratic Loss	0.01, 0.008
Long Short-Term Memory (LSTM-CFC)	Softmax	0.02, 0.015
Long Short-Term Memory (LSTM-CRF)	Squared Loss	10.5, 10.2
Bidirectional-RNN	Softmax	0.6, 0.5
Recurrent Network (RNN)	Softmax	0.55, 0.5
Dynamic RNN	Softmax	0.15, 0.1

System Evaluation

- **Evaluation scenarios:**

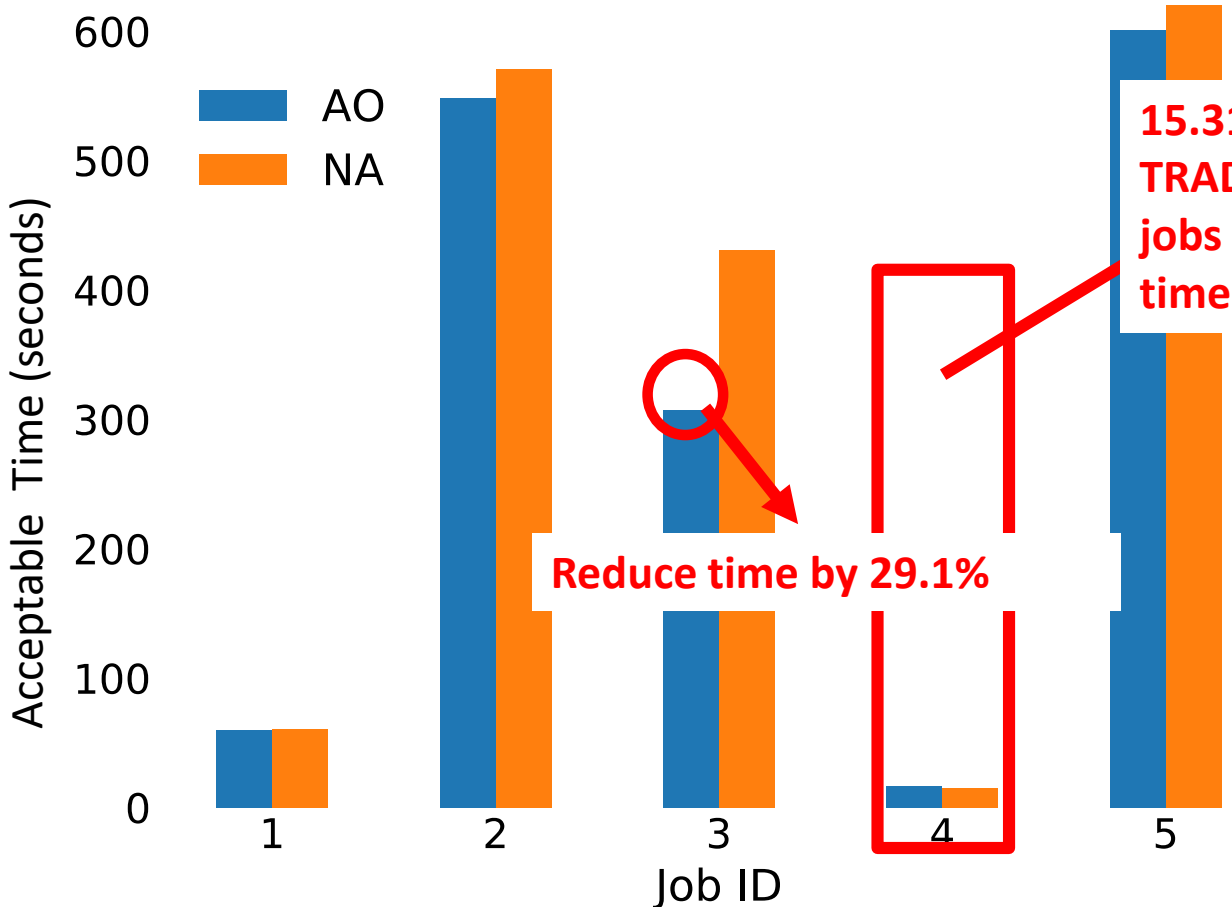
  1. **Acceptable Only**: < Acceptable, _> (denoted as **AO**)
  2. **Full two-tier Target**: < Acceptable, Objective> (denoted as **FT**)
  3. **No Algorithm**: <_, _> (denoted as **NA**)

- **Evaluation metrics:**

  1. **Acceptable and Objective time**: the time cost that each job achieve the two-tier target.
  2. **CPU Usage**: the CPU usage for the deep learning applications

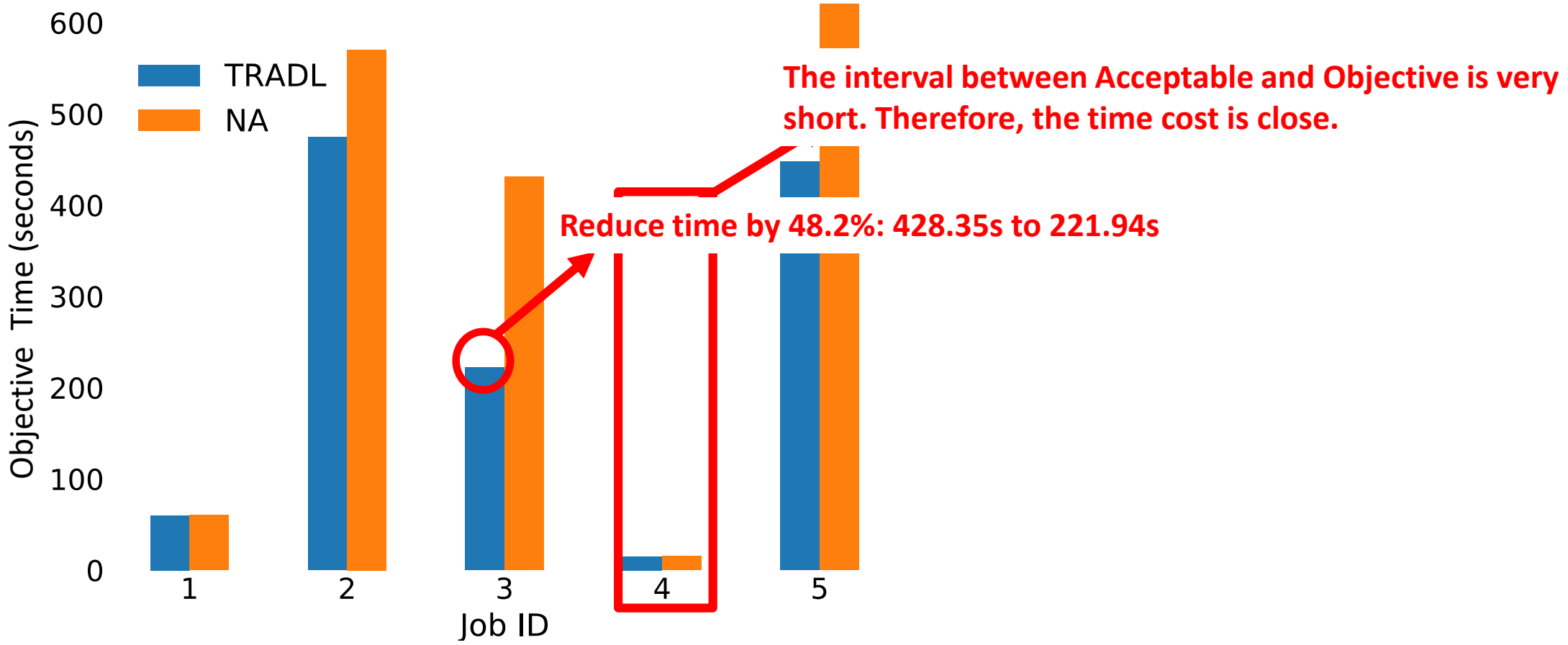
Acceptable and Objective time

- Acceptable Only

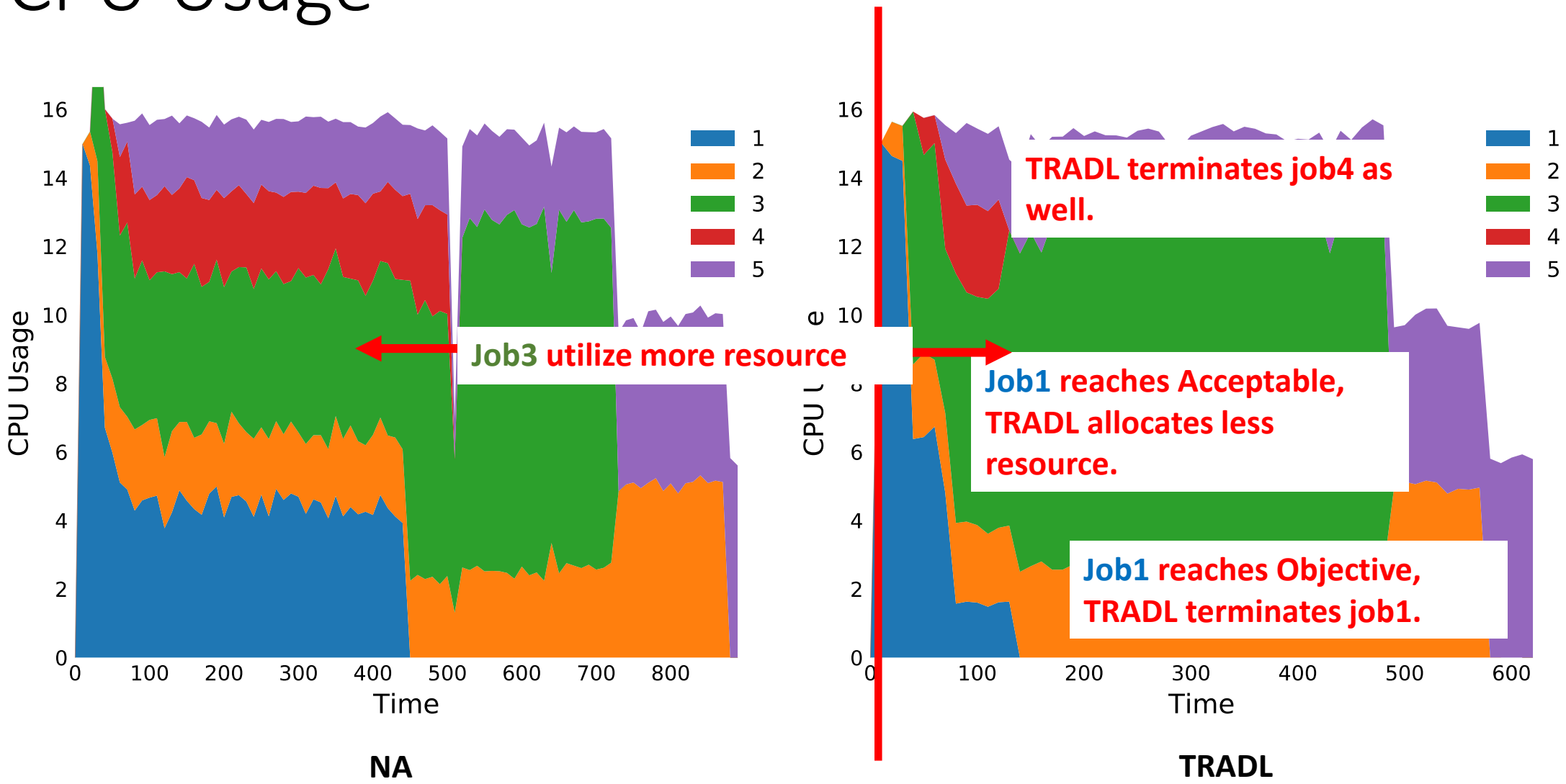


Acceptable and Objective time

- Full two-tier Target

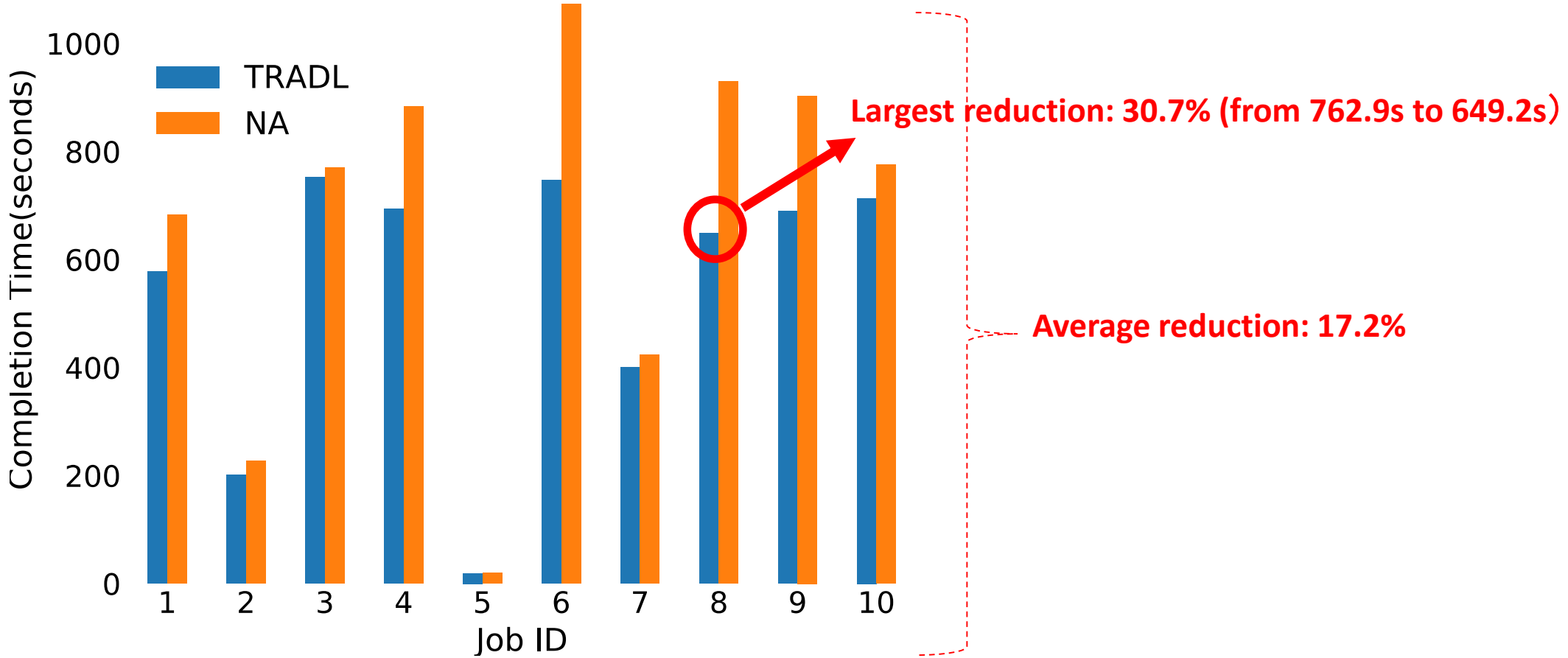


CPU Usage


CPU Usage
Time
1
2
3
4
5
Job3 utilize more resource
TRADL terminates job4 as well.
Job1 reaches Acceptable, TRADL allocates less resource.
Job1 reaches Objective, TRADL terminates job1.
NA
TRADL

Scalability

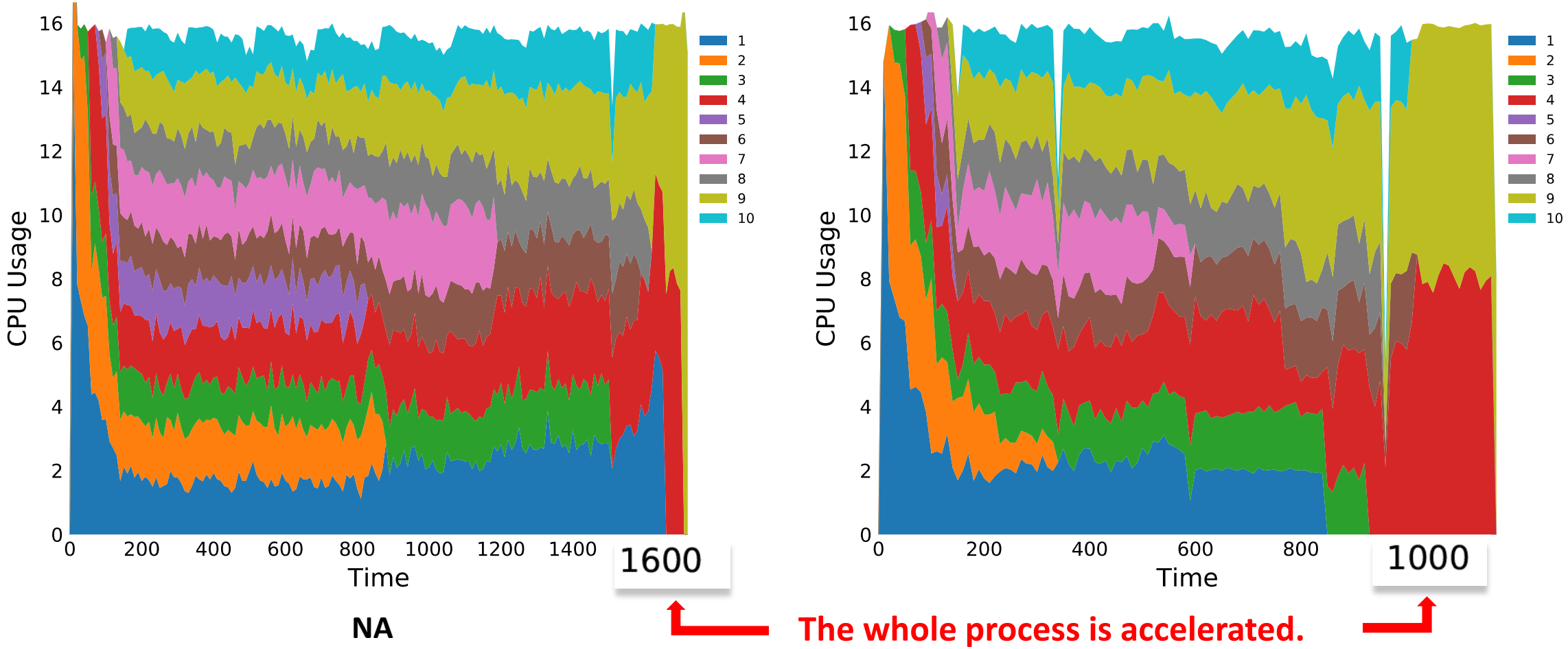
- Acceptable and Objective time



Scalability

- CPU Usage

Job4 and Job9 utilize more resource than NA.



Outline

- Introduction & Background
- Motivation
- *TRADL* Modules
- Evaluation
- Conclusion

Conclusion

- Based on the assumption that users can tolerate a certain level of error rate, we develop *TRADL*, which dynamically updates the resources configurations while the jobs are running to accelerate the training process.

- *TRADL* shows a significant improvement on the time cost of achieving the targets, and the largest reduction is 48.2%.

Thank you!
Q&A

if $RU(c_i) > \frac{1}{n}$ **then**

c_i.set_limits($\frac{1}{n \times 2}$)

else if $RU(c_i) \leq \frac{1}{n}$ **then**

c_i.set_limits($\frac{1}{n+1}$)