Project page

JUNE 18-22, 2023
CVPR
VANCOUVER, CANADA
WED-AM-352

EcoTTA: Memory-Efficient Continual Test-time Adaptation via Self-distilled Regularization

Junha Song[2] Jungsoo Lee[1] In So Kweon[2] Sungha Choi[1]

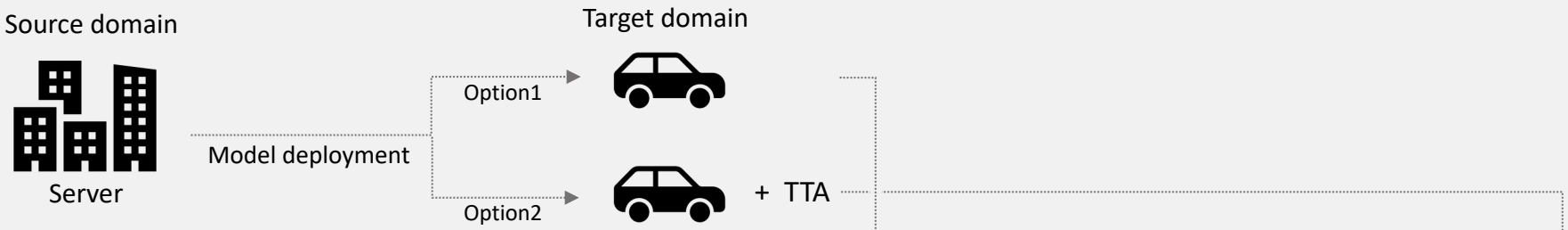
[1]Qualcomm AI Research [2]KAIST

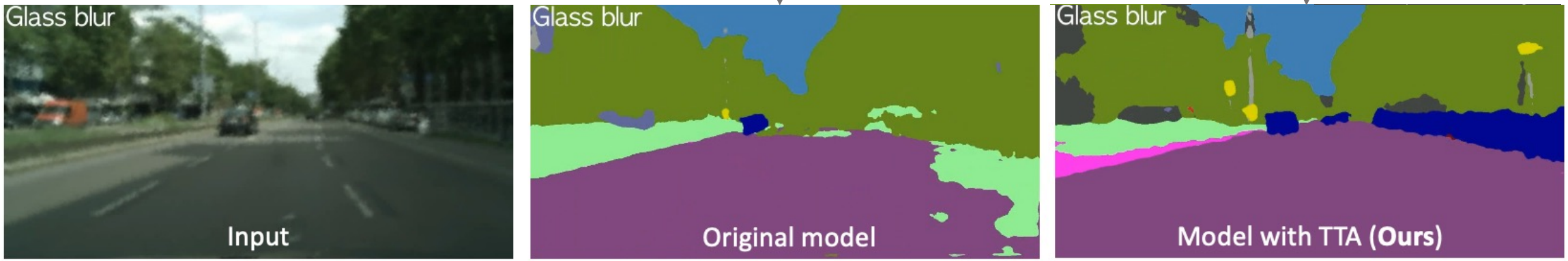
Qualcomm KAIST

Overview (1/2)

Test-time Adaptation

🤔 **Test-time adaptation (TTA)** is a cutting-edge AI capability that allows a deployed model to **adapt** itself to a new environment **during the testing phase**.



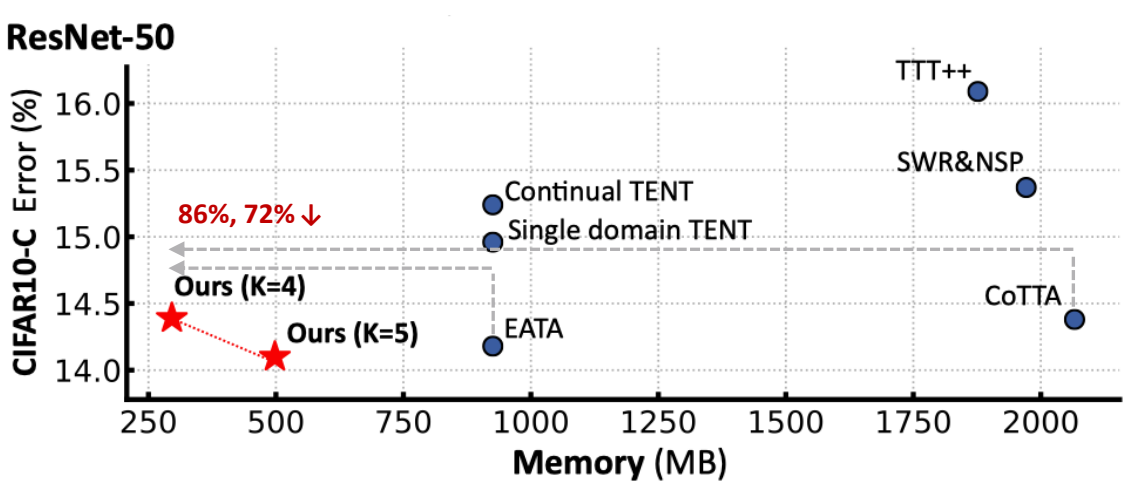


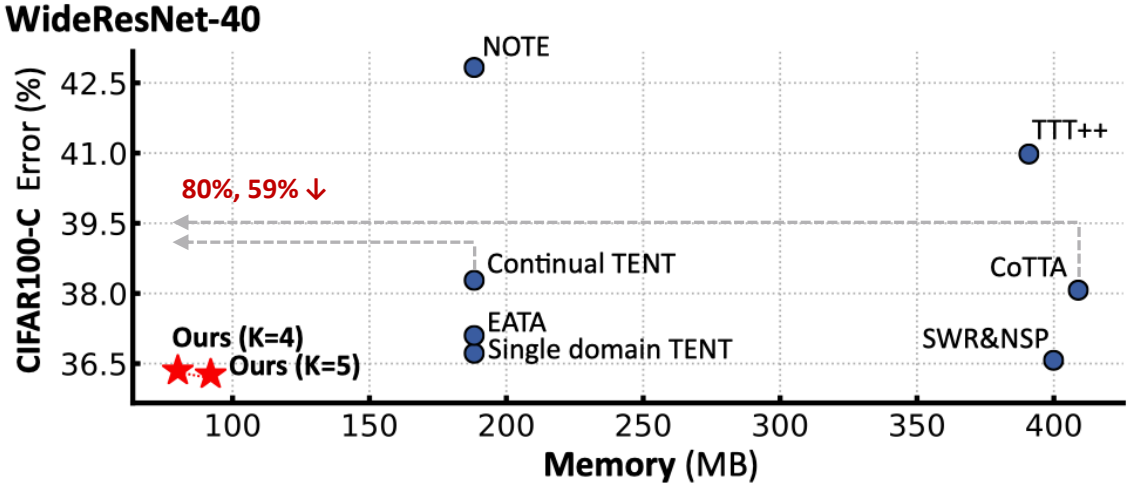
Overview (2/2)

EcoTTA | Memory-Efficient Continual Test-time Adaptation via Self-distilled Regularization

😮 Our work aims to make TTA **practical and applicable** in edge devices (e.g., robots or autonomous vehicles).

🤠 We design **memory-efficient architecture** which minimizes memory usage (i.e., activations) by up to 86% compared to state-of-the-art methods. Moreover, **our novel regularization** prevents overfitting by leveraging the knowledge acquired during pre-training, which is distilled from the frozen original model.

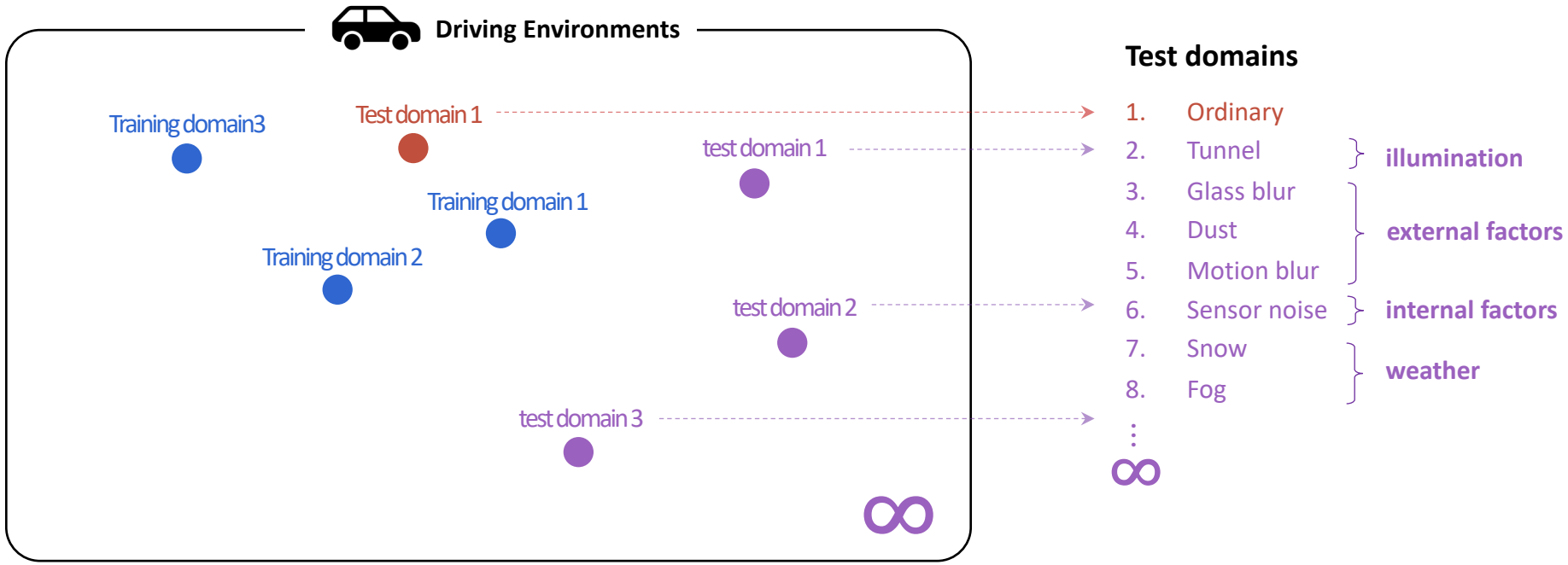




Introduction (1/5)

Robustness of Neural Networks

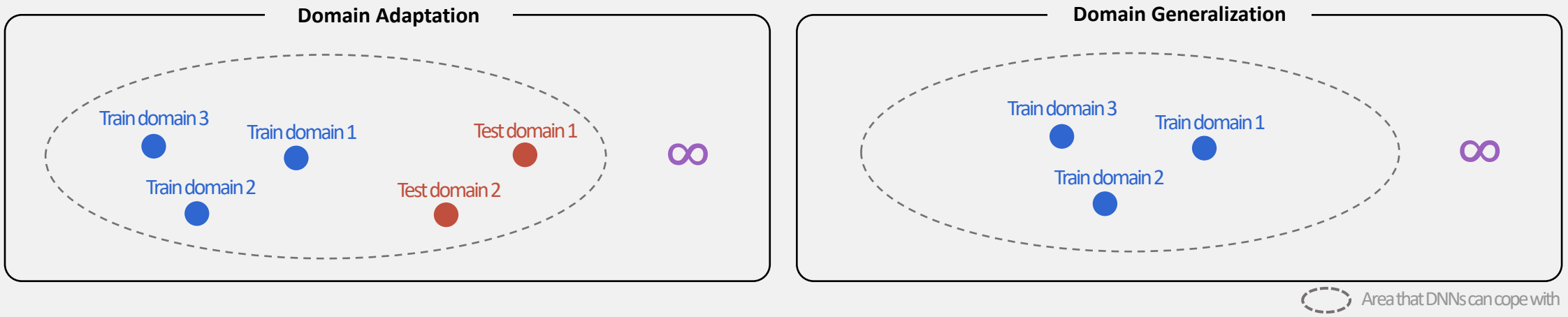
- Deep neural networks (DNNs) have good performance on **test domain** similar to train domain.
- But DNNs often suffer from **poor performance** on **test domain** significantly different from train domain.



Introduction (2/5)

Mitigating Domain Shift

- Several research fields attempt to address this problem,
  1) **Domain adaptation (DA)**
     - DA adapts the test domain using both training data and test data during pretraining stage.
  2) **Domain generalization (DG)**
     - DG learns invariant representation with only training data.

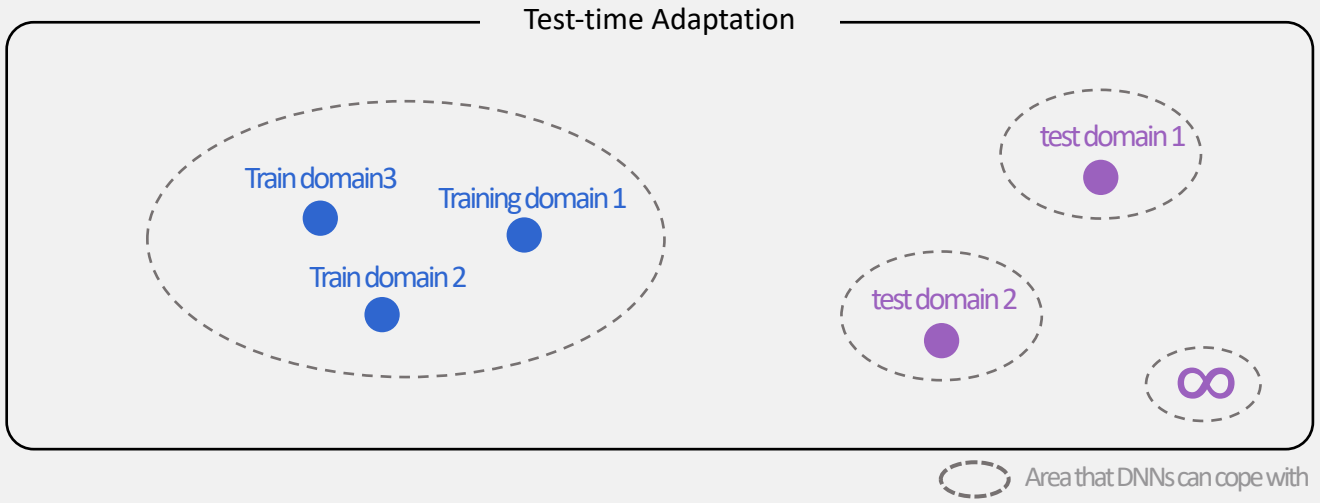


However, both DA and DG can not cope with **infinite test domains**.

Introduction (3/5)

Test-time Adaptation

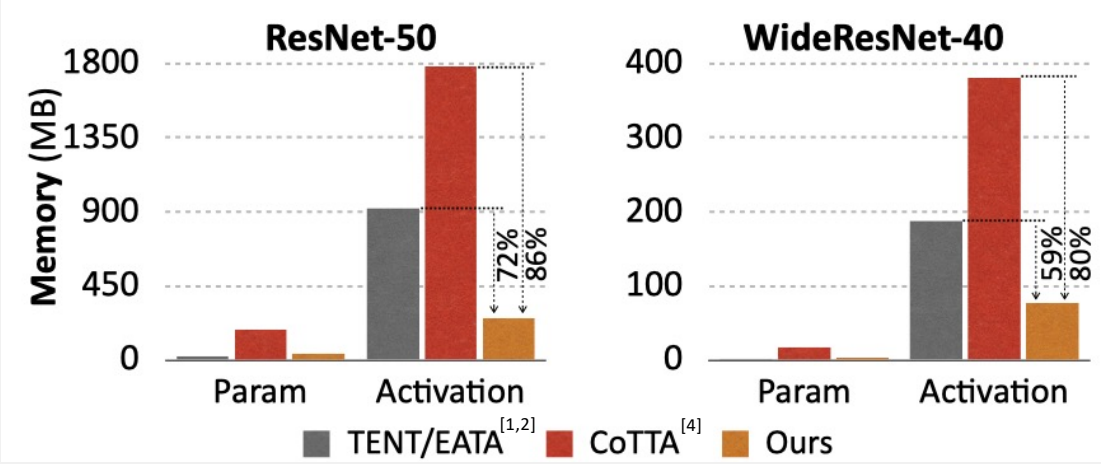
- The **TTA** approach overcomes the domain shift by **directly adapting to test domain**, Instead of enhancing generalization ability during the training time (such as DG).



Introduction (4/5)

Motivation 1

- TTA is conducted in edge devices (eg, robots or autonomous vehicles) which are likely to be memory-constrained.
- **Reducing memory usage** is crucial but has been overlooked in previous TTA studies [1,2,3,4].

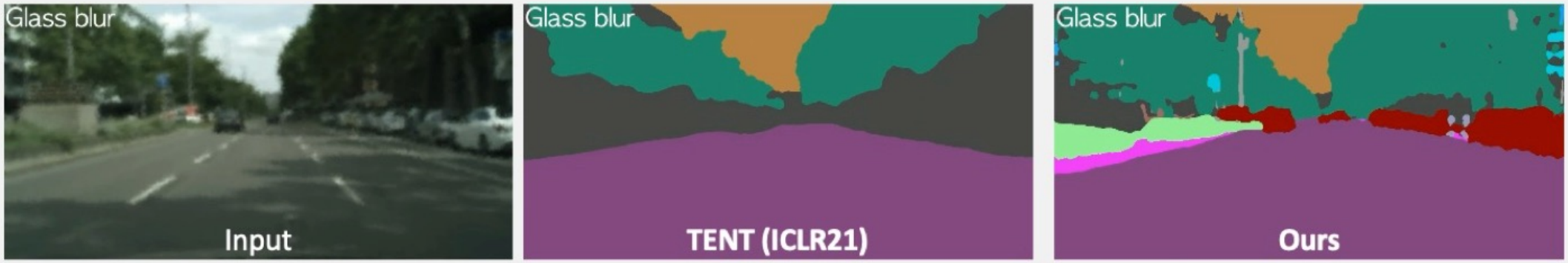


[1] Tent: Fully test-time adaptation by entropy minimization. ICLR, 2021.
[2] EATA: Efficient test-time model adaptation without forgetting. ICML, 2022.
[3] NOTE: Robust continual test-time adaptation via instance-aware BN. NeurIPS, 2022.
[4] CoTTA: Continual Test-time adaptation. CVPR, 2022.

Introduction (5/5)

Motivation 2

- Existing TTA [1,2,3,4] methods without regularization eventually face **overfitting** in **long-term adaptation** due to the effect of catastrophic forgetting and error accumulation.



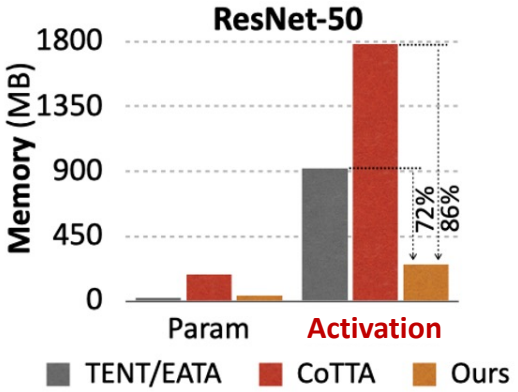
- **Previous challenges**
  1) **High computation overhead**
  2) **Overfitting in long-term adaptation**

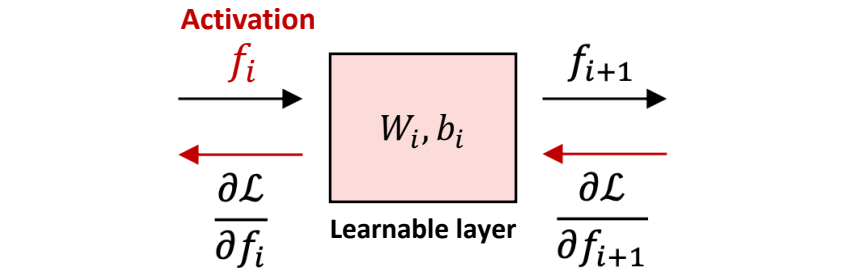
- **Our goal is to develop applicable and practical TTA approach.**
  1) **Memory-efficient architecture for TTA**
  2) **Self-distilled regularization**

[1] Tent: Fully test-time adaptation by entropy minimization. ICLR, 2021.
[2] TTT++: When does self-supervised test-time training fail or thrive? NeurIPS, 2021.
[3] SWR&NSP: Improving test-time adaptation via shift-agnostic weight regularization. ECCV, 2022.
[4] Contrastive test-time adaptation. CVPR, 2022.

Approach (1/4)

Prerequisite





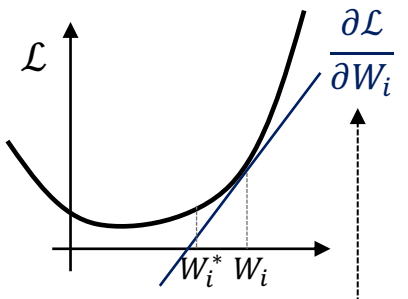
Forward

$$f_{i+1} = W_i \cdot f_i + b_i$$

Backward

$$\frac{\partial \mathcal{L}}{\partial f_i} = W_i^T \cdot \frac{\partial \mathcal{L}}{\partial f_{i+1}}$$

$$\frac{\partial \mathcal{L}}{\partial W_i} = f_i^T \cdot \frac{\partial \mathcal{L}}{\partial f_{i+1}}$$

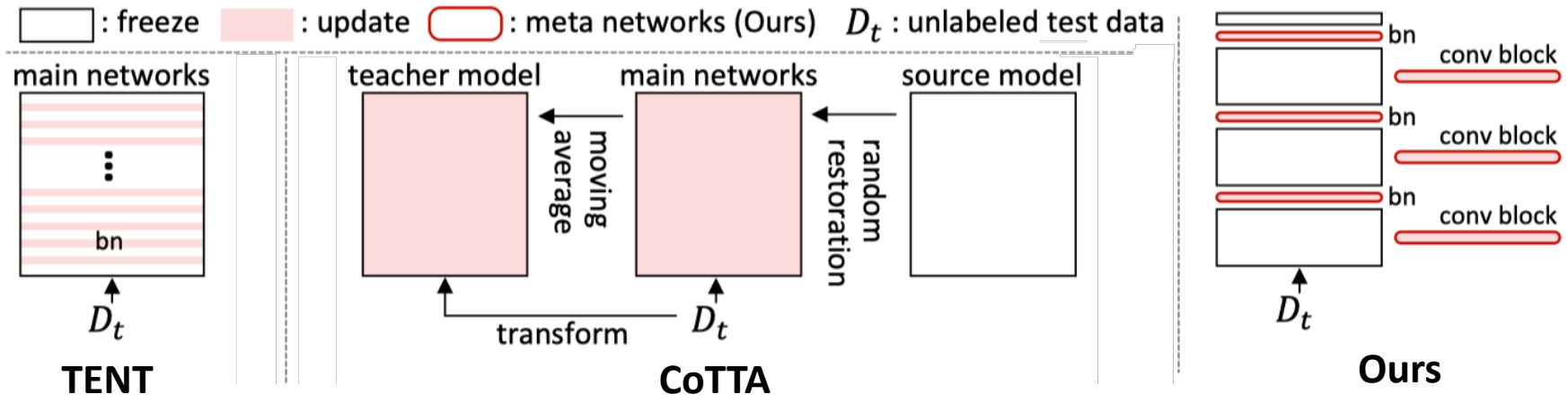


$$W_i^* = W_i - Lr \cdot \frac{\partial \mathcal{L}}{\partial W_i}$$

- TTA works update model parameters to adapt to the target domain. This process inevitably requires additional memory to store the **activations** which refer to the intermediate features stored during the forward propagation.
- Note that only learnable layers, not frozen layers, must store the activations.

Approach (2/4)

Overview of Our Approach



A. TENT updates ***multiple*** **batch norm** layers, in which large activations must be stored for gradient calculation.

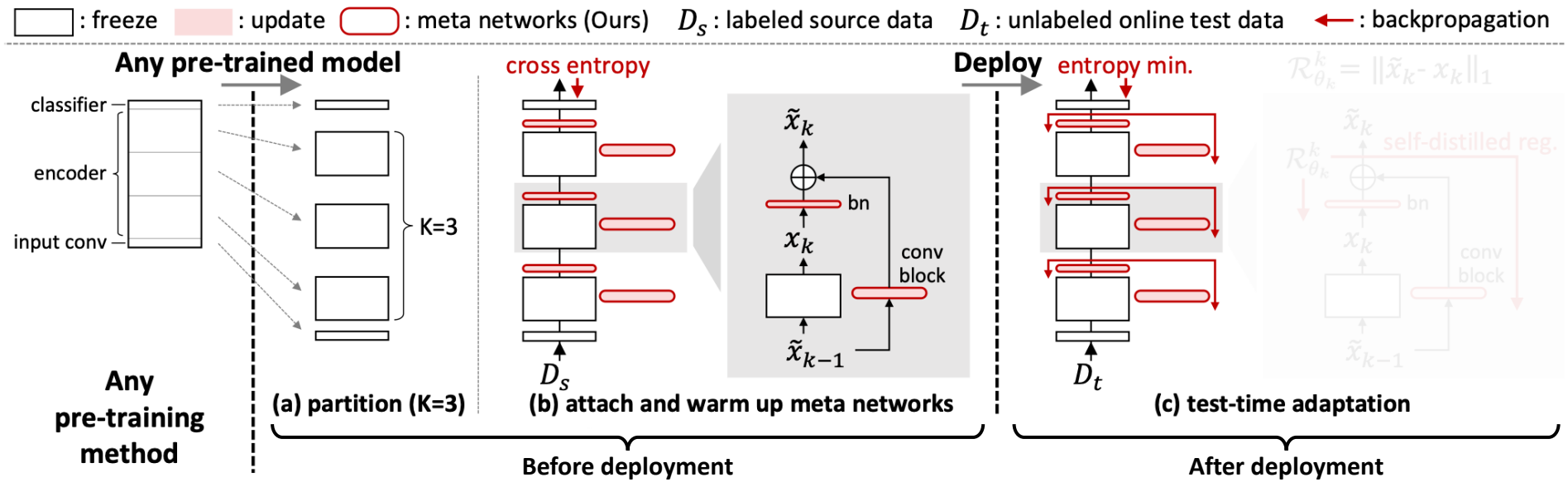
B. In CoTTA, an ***entire*** **network** is trained with additional strategies for continual adaptation that requires a significant amount of both memory and time.

C. In contrast, our approach requires a minimum size of activations by updating ***only a few*** **layers**. Also, stable long-term adaptation is performed by our proposed *regularization*.

Approach (3/4)

Memory-efficient Architecture

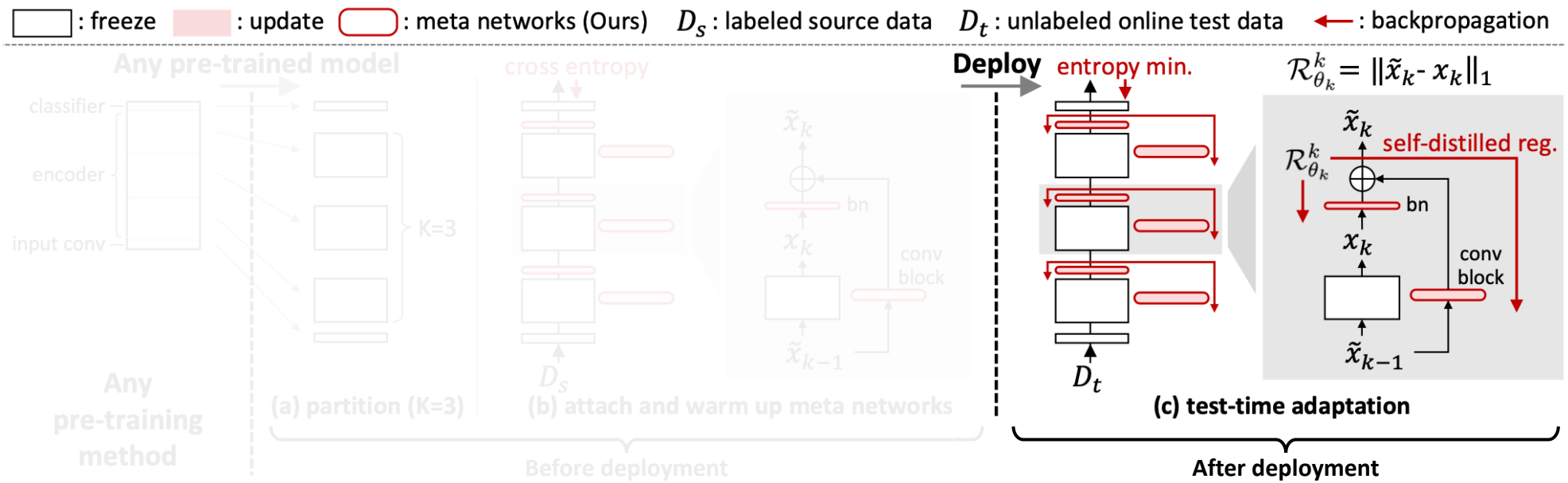
Video



- Our approach only updates the newly added **meta networks** on the target domain. The steps are the following:
  - i. **Before deployment**, we take any type of pre-trained model.
  - ii. The encoder of the pre-trained model is divided into K parts. We partition the shallow parts of the encoder more (i.e., densely) compared to the deep parts of it.
  - iii. The **meta networks** are **attached** to each part of the original networks and warmed up with train dataset.
  - iv. **After deployment,** only the meta networks are updated with unsupervised loss while the original networks are frozen.

Approach (4/4)

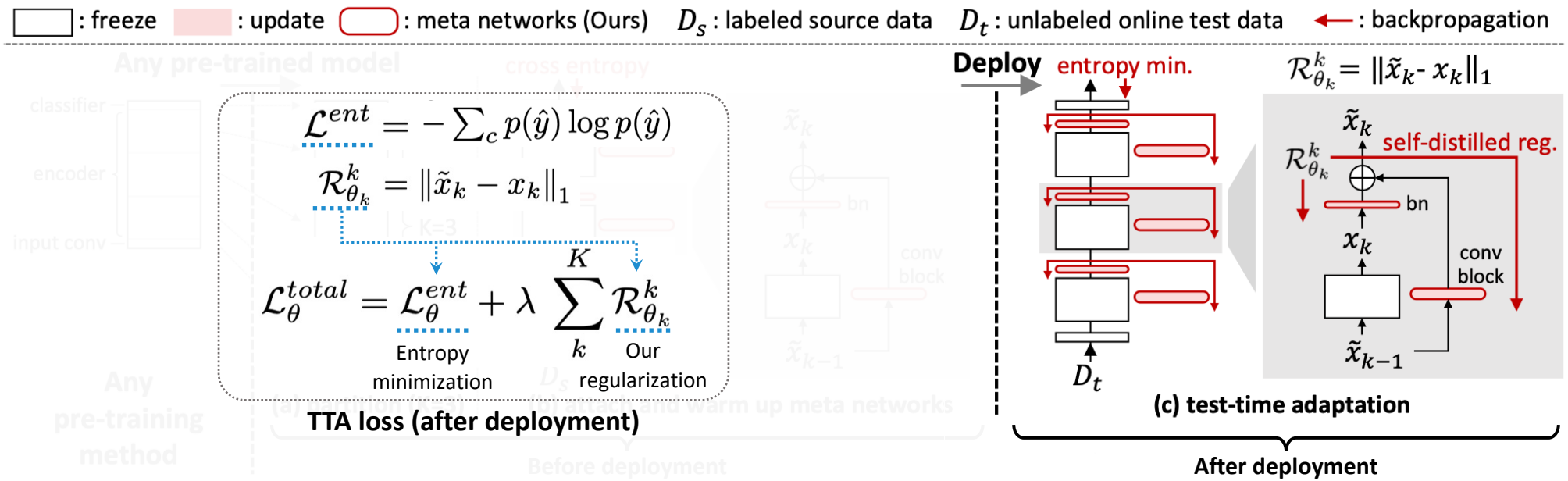
Self-distilled Regularization



- We regularize the output $\tilde{x}_k$ of the meta networks not to deviate from the output x_k of the frozen original networks.
- The output x_k of the *frozen* original networks contains the *knowledge of the train domain* consistently.
- We can **prevent *catastrophic forgetting*** by maintaining the source domain knowledge and ***error accumulation*** by utilizing the class discriminability of the original model.

Approach (4/4)

Self-distilled Regularization

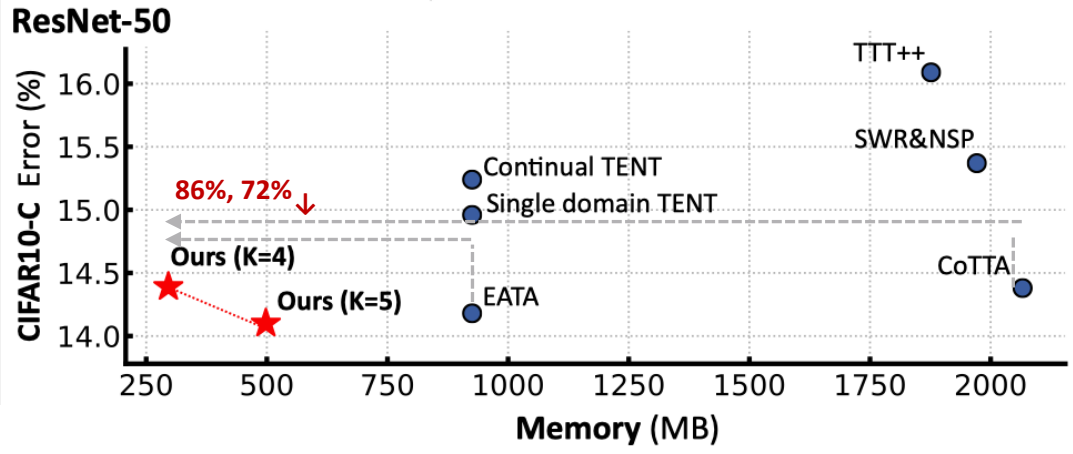


- We regularize the output $\tilde{x}_k$ of the meta networks not to deviate from the output x_k of the frozen original networks.
- The output x_k of the *frozen* original networks contains the *knowledge of the train domain* consistently.
- We can **prevent *catastrophic forgetting*** by maintaining the source domain knowledge and ***error accumulation*** by utilizing the class discriminability of the original model.

Results (1/2)

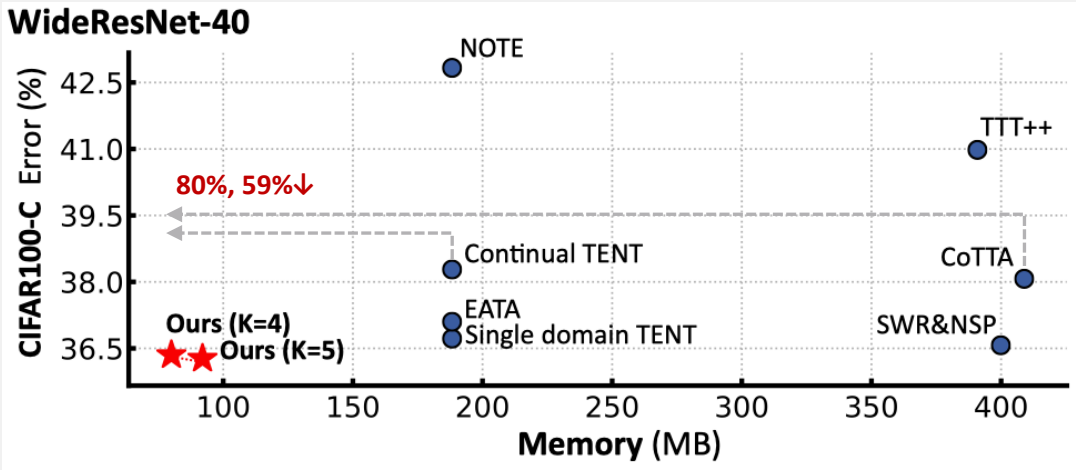
Image Classification

(a) Comparison of average error rate (%) on **CIFAR10-C**



K: Model partition factor

(b) Comparison of average error rate (%) on **CIFAR100-C**



(c) Comparison of average error rate (%) on **ImageNet-C**

	ResNet-50 (AugMix)	
Method	Avg. err ↓	Memory (MB)
Source	74.36	91
BN Stats Adapt	57.87	91
Continual TENT (ICLR21)	56.1	1486
EATA (ICLM22)	54.9	1486
CoTTA (CVPR22)	54.6	3132
Ours (K=4)	55.2	**438** (86, 72%↓)
Ours (K=5)	**54.4**	747 (75, 51%↓)

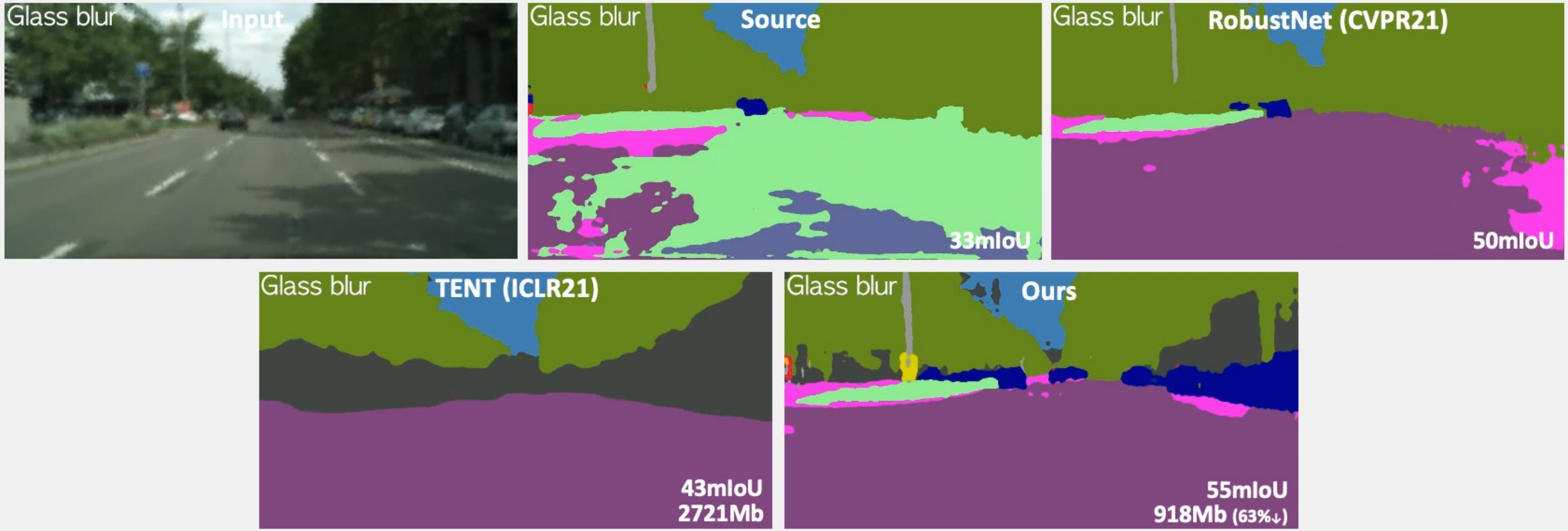
	WideResNet-40 (AugMix)		ResNet-50	
Method	**Avg. err ↓**	**Mem. (MB)**	**Avg. err ↓**	**Mem. (MB)**
Source	69.7	11	73.8	91
BN Stats Adapt	41.1	11	44.5	91
Single do. TENT (ICLR21)	36.7	188	40.1	926
Continual TENT (ICLR21)	38.3	188	45.9	926
TTT++ (NeurIPS21)	41.0	391	44.2	1876
SWR&NSP (ECCV22)	36.6	400	44.1	1970
EATA (ICML21)	37.1	188	39.9	926
CoTTA (CVPR22)	38.1	409	40.2	2064
Ours (K=4)	36.4	80 (80, 58%↓)	39.5	296 (86, 68%↓)
Ours (K=5)	**36.3**	92 (77, 51%↓)	**39.3**	498 (76, 46%↓)

Results (2/2)

Semantic Segmentation

Video

DeepLabV3 with ResNet50



Conclusion | Memory-**E**fficient **Co**ntinual **T**est-**t**ime **A**daptation via Self-distilled Regularization

- We propose a simple yet effective approach that improves TTA performance and saves a significant amount of memory.
- First, we presented a memory-efficient architecture which minimizes the intermediate activations used for gradient calculations.
- Second, we proposed self-distilled regularization to prevent overfitting during long-term adaptation.
- We verified the memory efficiency and TTA performance of our approach with extensive experiments on diverse datasets and backbone networks.

Thank you

Qualcomm

Follow us on:

For more information, visit us at:

qualcomm.com & qualcomm.com/blog