ICLR 2023

TTN: A Domain-Shift Aware Batch Normalization in Test-Time Adaptation

Hyesu Lim
KAIST

Byeonggeun Kim

Jaegul Choo
KAIST

Sungha Choi
Qualcomm Korea YH

Qualcomm

KAIST AI
Kim Jaechul Graduate School

Introduction

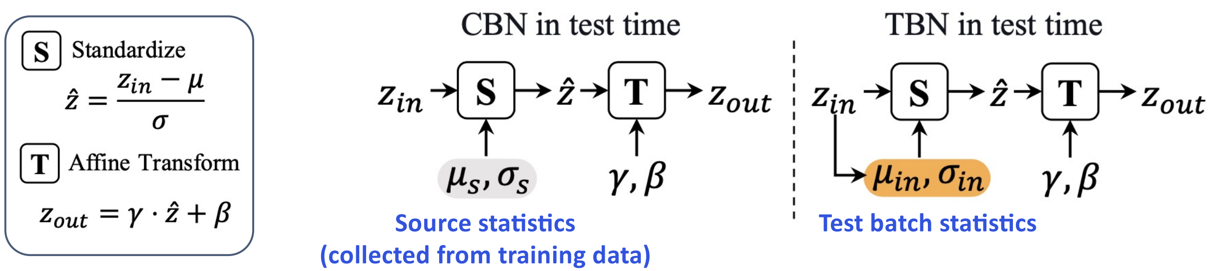
Background: Test-Time Adaptation (TTA)

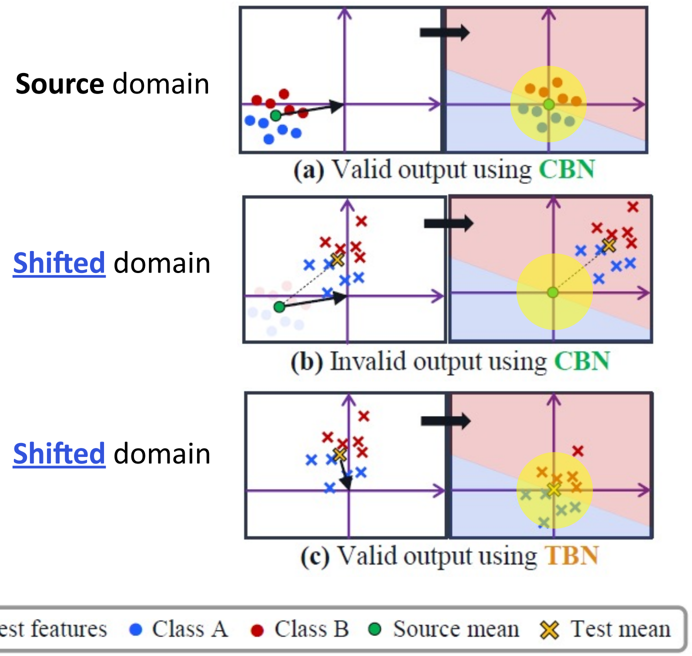
- When train (source) and test (target) domains differ, i.e., **domain shift** from source to target, deep neural networks (DNNs) suffer from **performance degradation**
- **Test-time adaptation (TTA)** aims to overcome this problem

Introduction

Background: Test-Time Adaptation (TTA)

- Recent **test-time adaptation (TTA)** methods heavily depend on **transductive batch normalization (TBN)**
  - To overcome the weakness of **conventional batch normalization (CBN),** which is vulnerable to domain shifts
  - **TBN** uses test input statistics for standardization and is robust to the domain shifts



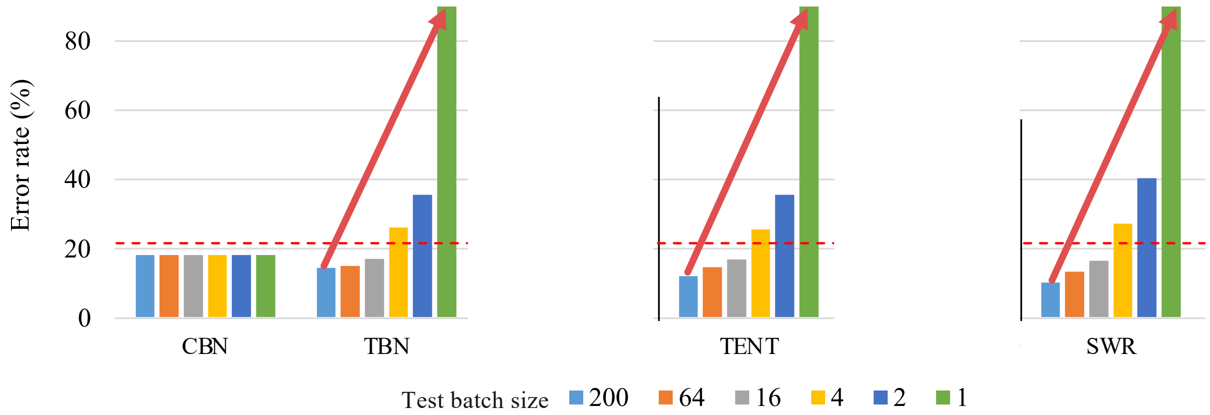


Introduction

Background: BN layers in TTA

- **TBN-based approaches**
  - **Strength**
    - Successfully **mitigate the domain-shift** between source and target
  - **Limitation**
    - Depend on **impractical assumptions**
      - Large test batch sizes (e.g., 200 or more), a single stationary target distribution
    - We observed that they suffer from **serious performance drop** when the assumptions are unsatisfied
      - Performance drops (i.e., error rate increase) in small test batch sizes

*** Spoiler**: Our proposed method (TTN) overcomes the dependency on impractical assumptions*



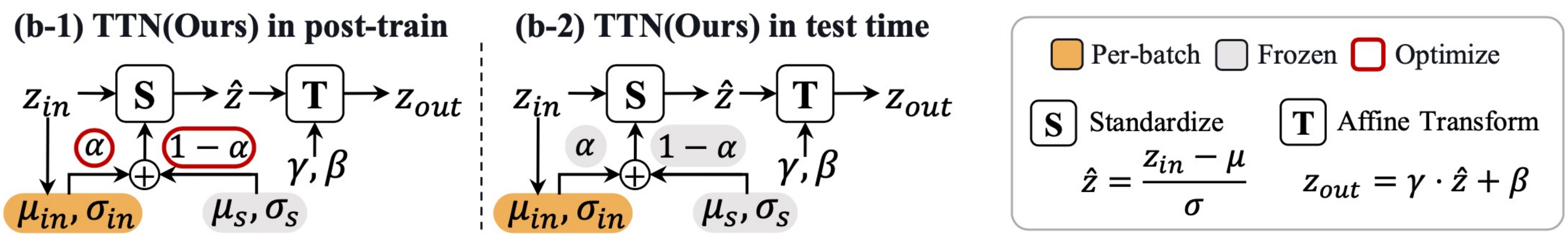
[1] TENT: Fully Test-Time Adaptation (ICLR'21)
[2] Improving Test-Time Adaptation via Shift-agnostic Weight Regularization and Nearest Source Prototypes (ECCV'22)

Proposed Method

Test-Time Normalization (TTN) Layer

- **Test-Time Normalization (TTN) layer**
  - **TTN** uses a interpolation of source and current test batch statistics using **learnable** interpolating weight α for standardization

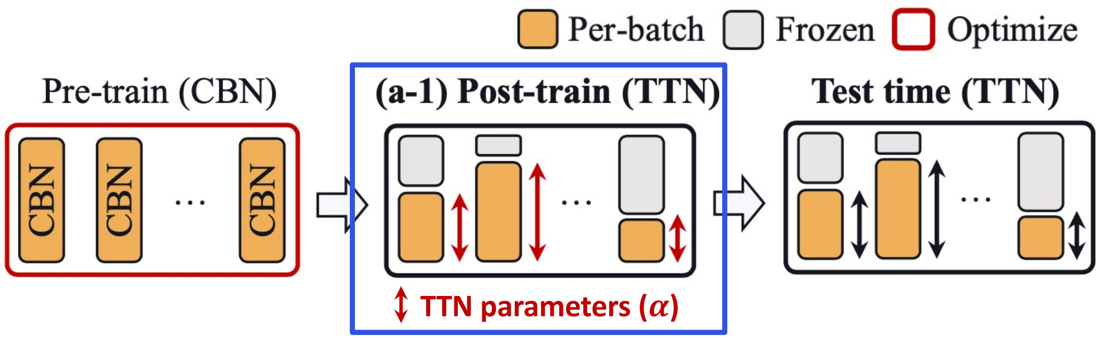
$$\tilde{\mu} = \alpha\mu + (1-\alpha)\mu_s, \quad \tilde{\sigma}^2 = \alpha\sigma^2 + (1-\alpha)\sigma_s^2 + \alpha(1-\alpha)(\mu-\mu_s)^2,$$



Proposed Method

Post-training

- **Post-training phase**
  - We **train the TTN parameter α** during a **post-training** phase (between pre-train and test time)
  - In post-training phase, all parameters are frozen except for α (i.e., only α is trained)



- We train α following model's **domain-shift sensitivity**
  - **Intuition**: we put more importance on **test batch statistics** when the model needs more **domain information**

Proposed Method

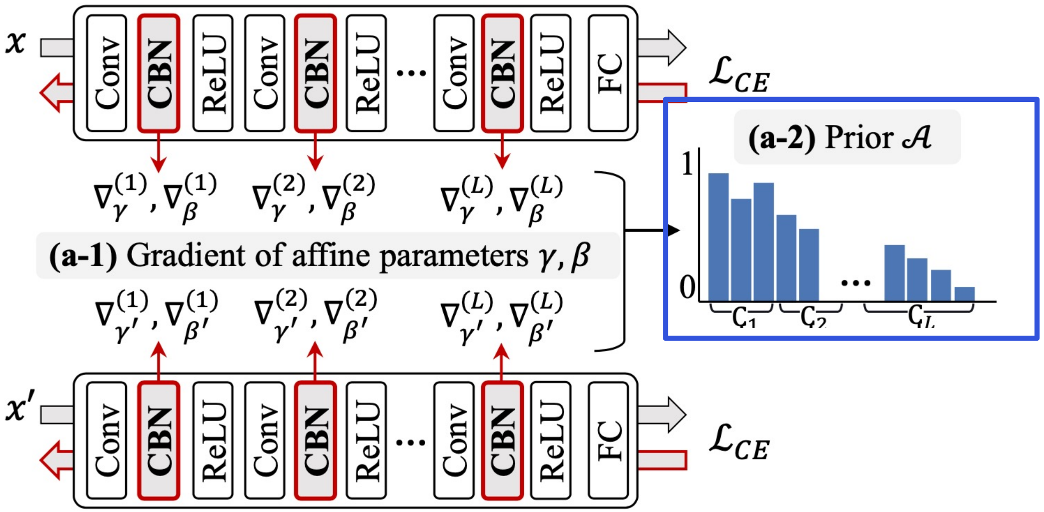
Post-training 1) Obtain Prior $\mathcal{A}$

- **Measuring model's domain-shift sensitivity using gradient distance score**
  - We measure the **difference between $\hat{\boldsymbol{z}}$ and $\hat{\boldsymbol{z}}'$** by **comparing the gradients** of the affine parameters γ and β
- **Intuition**
  - **Large difference** between $\hat{z}$ and $\hat{z}'$ means the layer (or channel) is **intensely affected by the domain shift**
    i.e., the layer (or channel) is handling **domain-related** knowledge

Proposed Method

Post-training 1) Obtain Prior $\mathcal{A}$

- **Measuring model's domain-shift sensitivity using gradient distance score**
  - We measure the **difference between $\hat{z}$ and $\hat{z}'$** by **comparing the gradients** of the affine parameters γ and β
  - We compute the **distance score** between two gradients and then define the **prior $\mathcal{A}$**



$$s = \frac{1}{N} \sum_{i=1}^{N} \frac{g_i \cdot g_i'}{\|g_i\| \|g_i'\|},$$ Cosine similarity

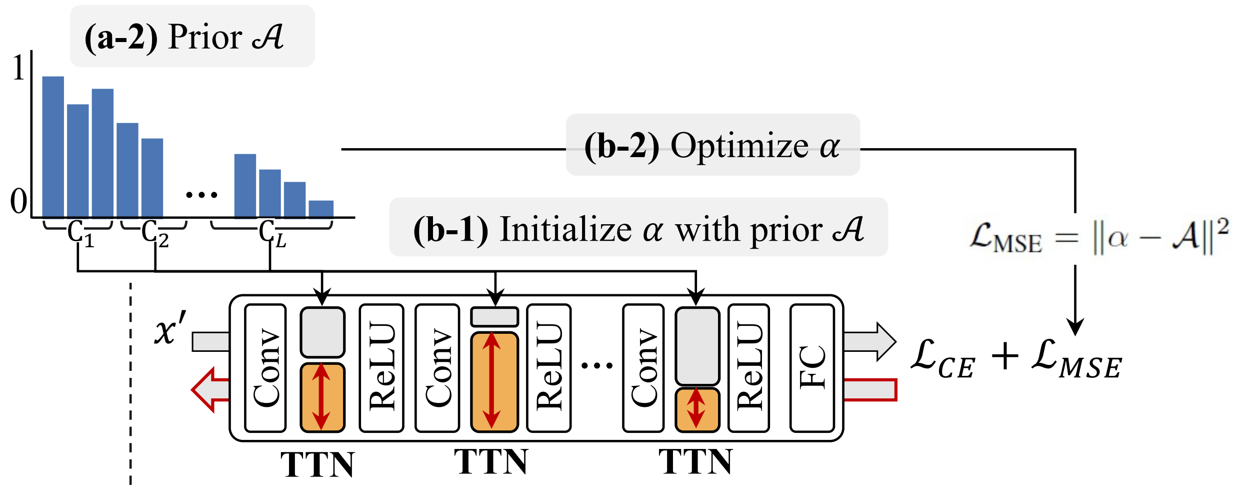
$$d^{(l,c)} = 1 - \frac{1}{2}\left(s_\gamma^{(l,c)} + s_\beta^{(l,c)}\right),$$ Distance score

$$\mathcal{A} = [d^{(1,\cdot)}, d^{(2,\cdot)}, \ldots, d^{(L,\cdot)}]^2,$$ Prior $\mathcal{A}$

Proposed Method

Post-training 2) Optimize α

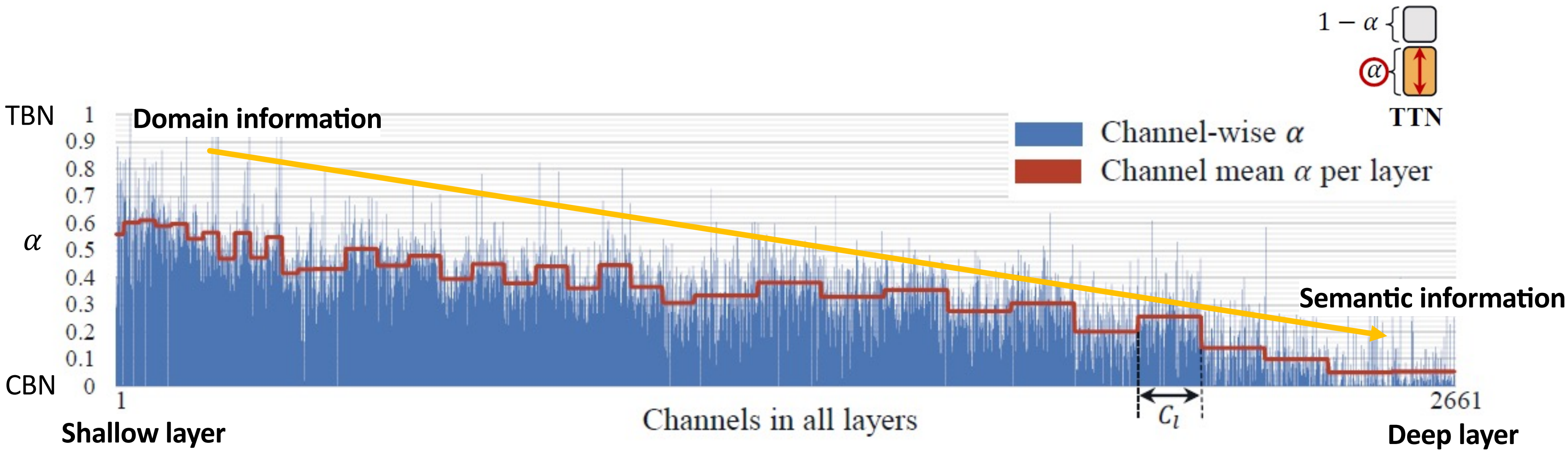
- **Optimize α**
  - Convert CBN layers to TTN layers
  - Initialize α with the **prior $\mathcal{A}$**
  - Optimize α with loss = $\mathcal{L}_{CE} + \mathcal{L}_{MSE}$
    - **Regularization with MSE loss**: to prevent α from moving too far from the initial point



Results

Visualization of the optimized α

- **Visualization of the optimized α**
  - We observed that the current test batch statistics are more used (i.e., α closer to 1) in shallower layers, where domain information is more dominant and vice versa



Results

Experimental results

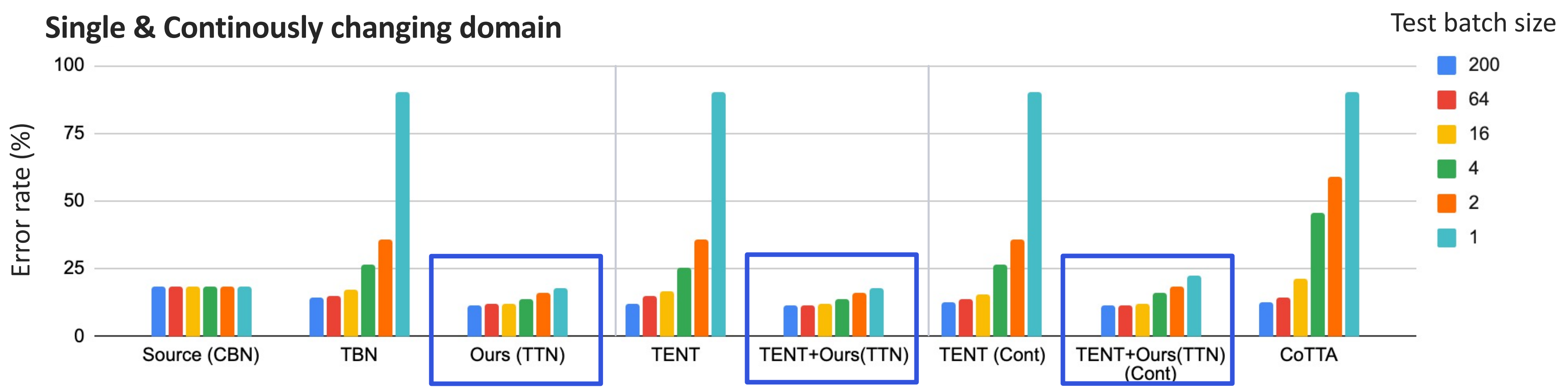
- **Experiments**
  - **Tasks**
    - Image classification (CIFAR-10-C, CIFAR-100-C, ImageNet-C)
    - Semantic segmentation (Cityscpase to BDD-100K, Mapiliary, GTA5, SYNTHIA)

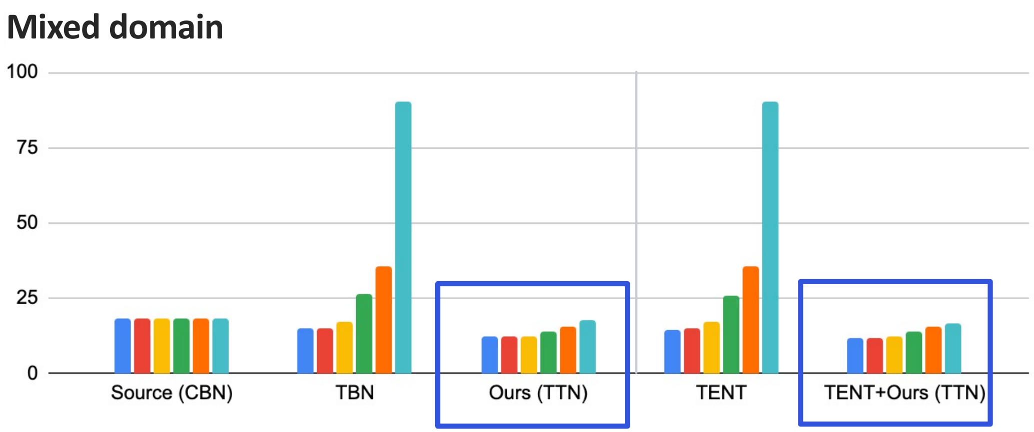
  - **Scenarios**
    1. **Single** domain adaptation scenario
       - Adapt to a single corruption type at a time. Reset model parameters whenever corruption type changes.
    2. **Continuously** changing domain adaptation scenario
       - Continuously update model to different corruption types without resetting.
    3. **Mixed** domain adaptation scenario
       - Single batch containing multiple corruption types
    4. **Class imbalanced** scenario
    5. Adaptation on **source domain** test samples (i.e., forgetting on source knowledge)

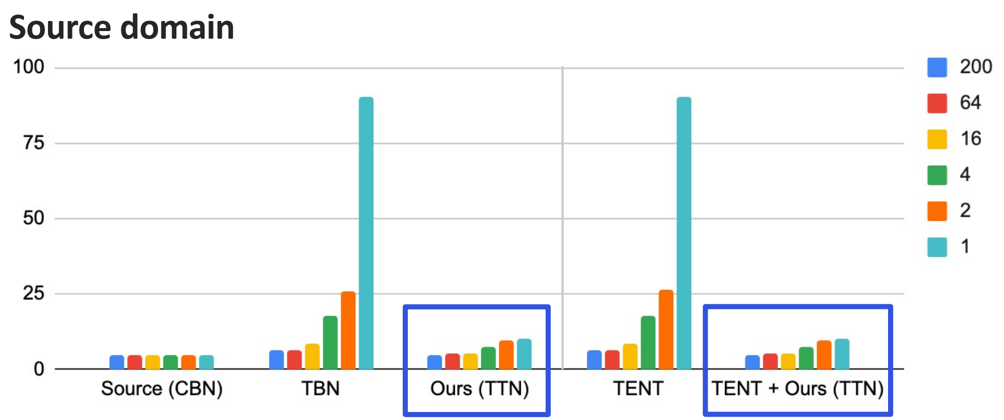
  - **Evaluation settings**
    - A wide range of **test batch sizes** (200, 64, 16, 4, 2, and 1)

Results

Experimental results (selected)







Summary

TTN: A DOMAIN-SHIFT AWARE BATCH NORMALIZATION IN TEST-TIME ADAPTATION

- **Task**
  - **Test-time adaptation (TTA)**, which aims to adapt models towards **test data** to overcome the performance degradation caused by **distribution shift**

- **Proposed Method**
  - **Test-time normalization (TTN) layer**, a new type of batch normalization layer, which combines **source** and **test batch statistics** using channel-wise interpolating weights considering the **sensitivity to domain shift**

- **Contribution**
  - TTN **flexibly adapts to new target domains** while **preserving the well-trained source knowledge**
  - TTN is **broadly applicable** to other TTA methods, since TTN does not alter training or test-time schemes (**backpropagation-free** adaptation)
  - TTN shows robust performance in **various practical scenarios**: a wide range of **test batch sizes (from 200 to 1)**, and three realistic evaluation scenarios: **stationary, continuously changing,** and **mixed** domain adaptation

Thank you

Follow us on:

For more information, visit us at:

qualcomm.com & qualcomm.com/blog