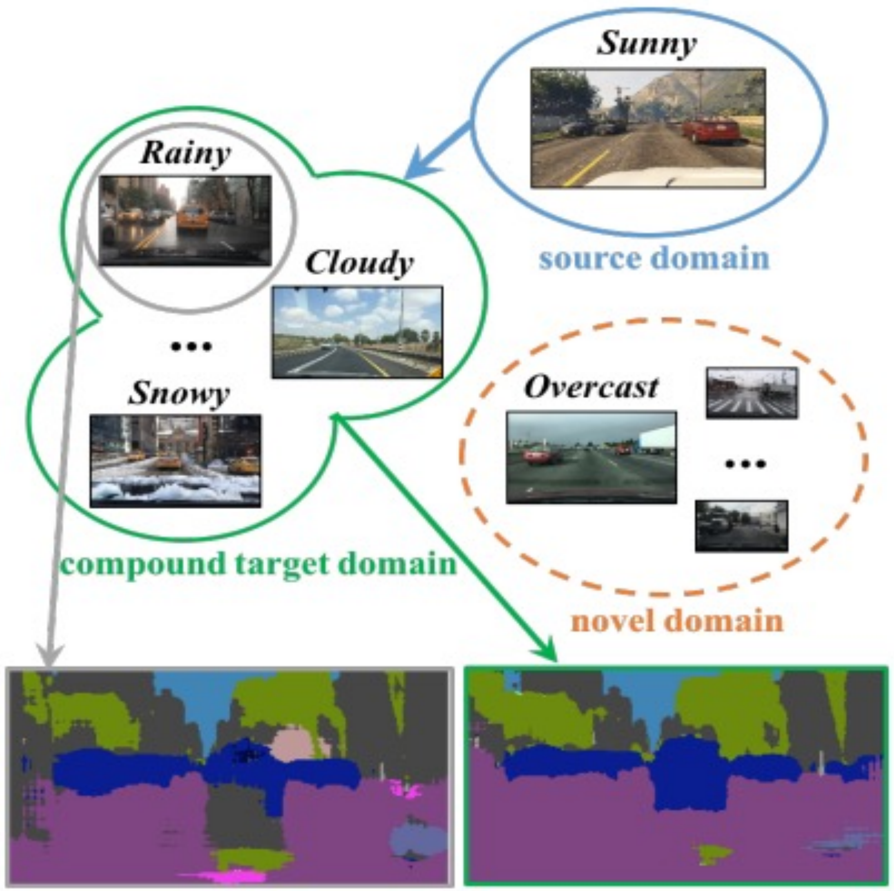
Feature-Based Instance Neighbor Discovery: Advanced Stable Test-Time Adaptation in Dynamic World

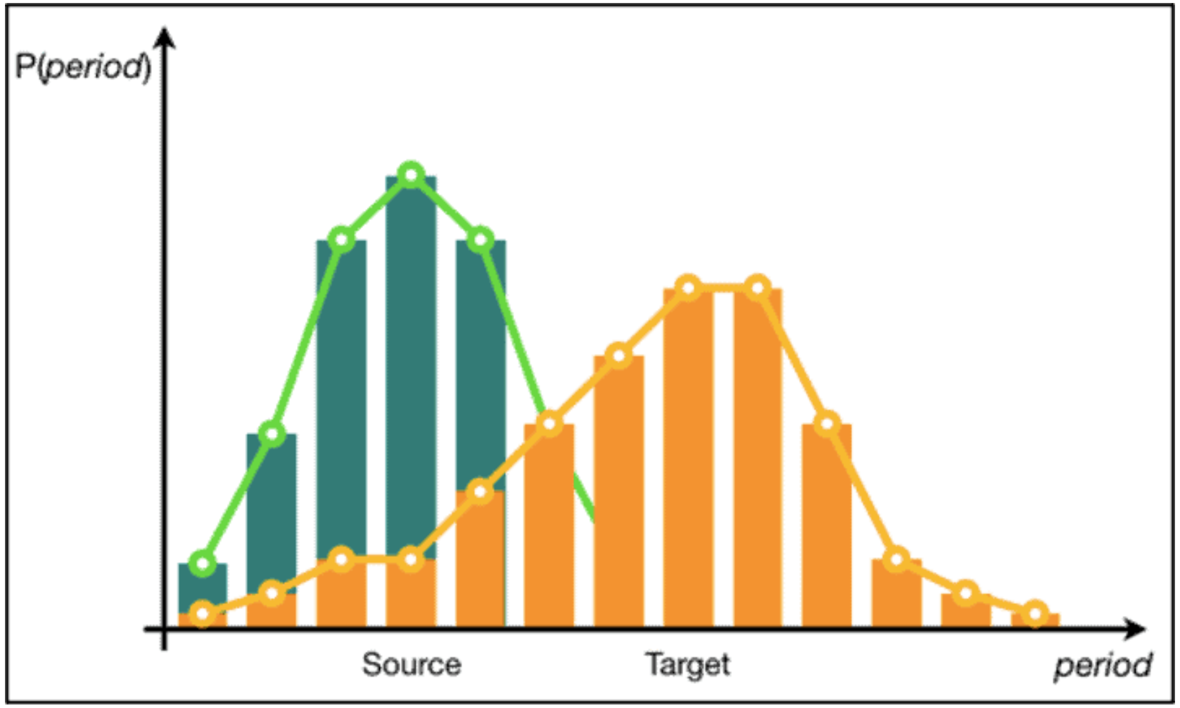
Qinting Jiang[1], Chuyang Ye[2], Dongyan Wei[3] , Bingli Wang[4] , Yuan Xue [1], Jingyan Jiang[5], Zhi Wang[1]

[1]Shenzhen International Graduate School, Tsinghua University
[2]Courant Institute of Mathematical Sciences, New York University
[3]CSDAI Department, Institut Polytechnique de Paris
[4] College of Information Engineering, Sichuan Agricultural University
[5]School of Artificial Intelligence, Shenzhen Technology University

Corresponding author: wangzhi@sz.tsinghua.edu.cn, jiangjingyan@sztu.edu.cn

Background



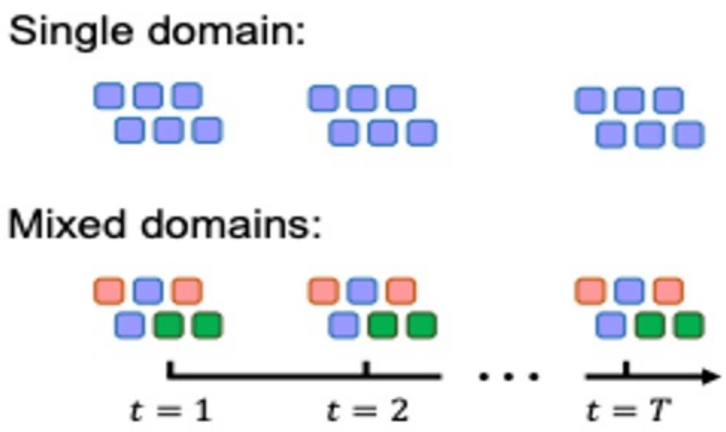


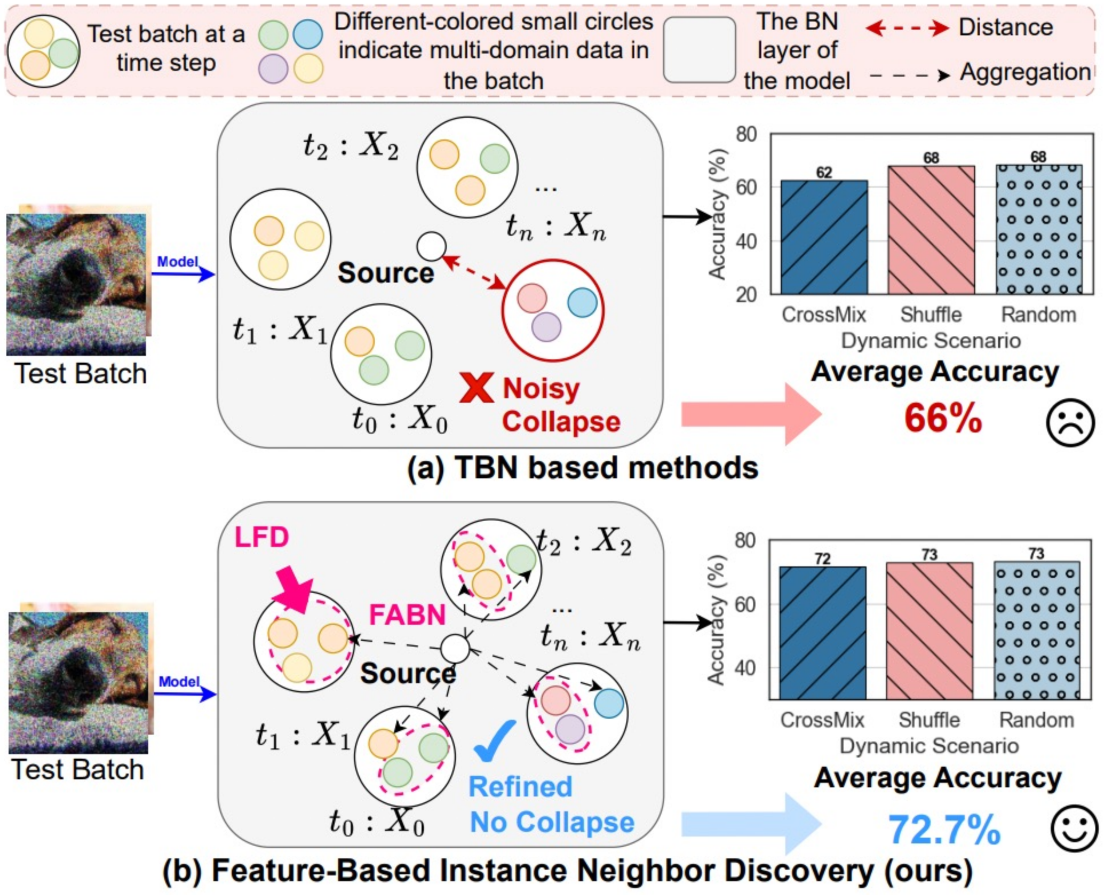
As the weather changes, the target domain gradually shifts, resulting in a decline in model's performance

Test-time adaptation (TTA) allows a model to adjust in real-time during inference, helping it maintain performance as test data changes.

Motivation

Dynamic scenarios: In real-world settings, test batches often consist of samples from one or multiple domains.





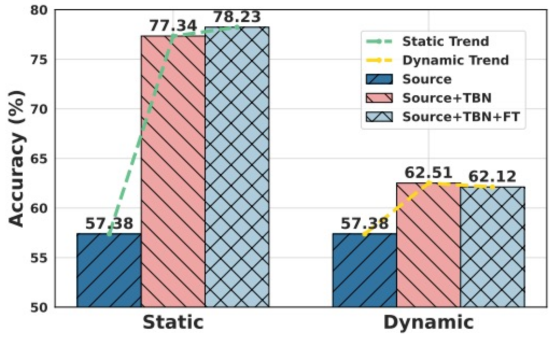
A "divide–and–conquer" test–time normalization strategy needs to be designed to avoid feature interference from multiple domains

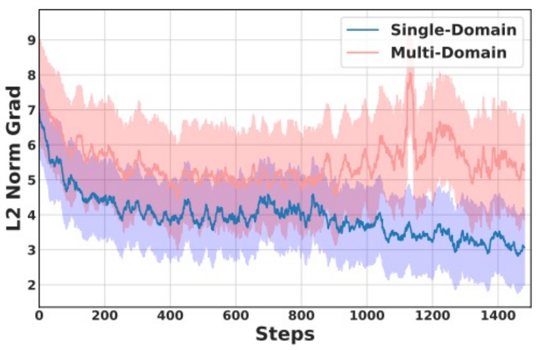
Observations

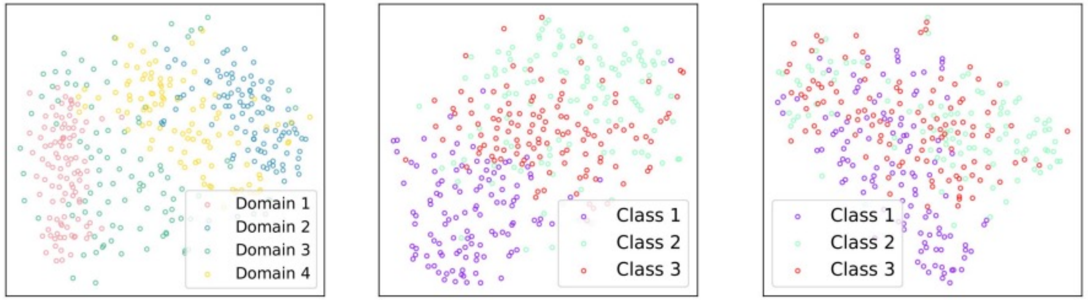
(1) Precise normalization is pivotal for effective TTA.

(2) A divide-and-conquer strategy is essential in dynamic environments.

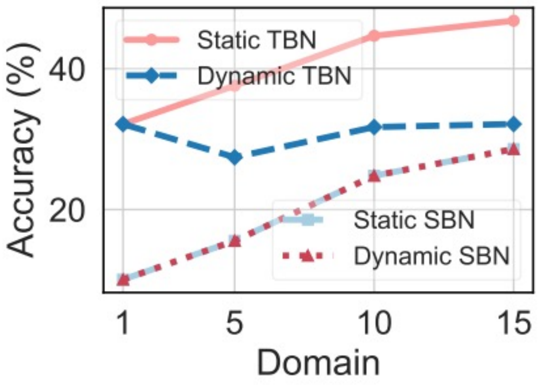
(3) Enhance normalization stability via generic knowledge from training data.

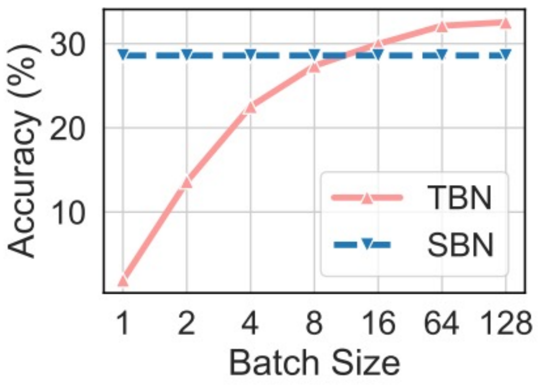






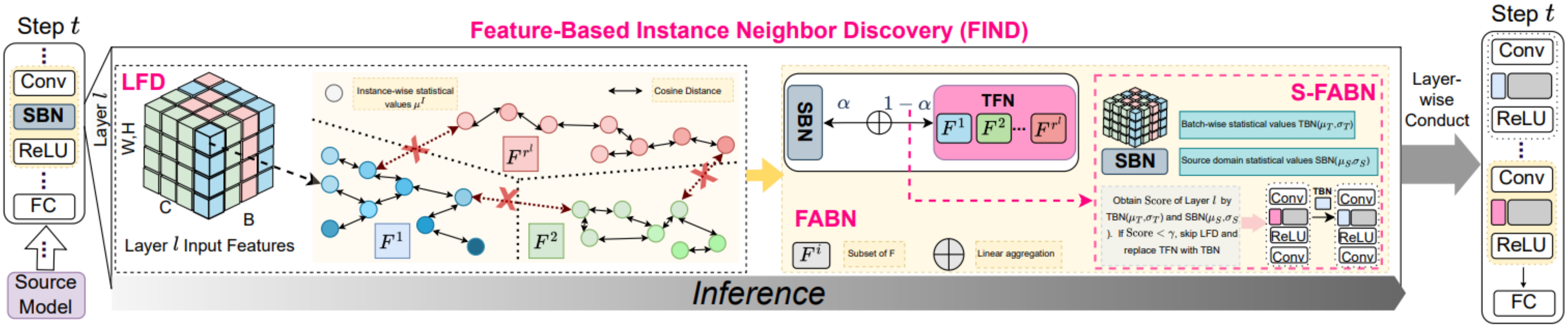
(a) Domain (Dynamic) (b) Class (Static) (c) Class (Dynamic)





Method

The proposed FIND involves three novel designs, including (1) layer-wise feature disentanglement (LFD), (2) feature-aware batch normalization (FABN) and (3) selective FABN (S-FABN).



LFD

(1) Purpose

LFD captures the distribution statistics of domain-relevant features by partitioning the features within each BN layer.

(2) Implementation

$$\mu_{i,c}^{I} = \frac{1}{L}\sum_{L} F_{i;c;L},$$

$$\mathrm{Sim}(i,j) = \frac{\mu_{i,c}^{I} \cdot \mu_{j,c}^{I}}{\left\|\mu_{i,c}^{I}\right\| \left\|\mu_{j,c}^{I}\right\|},$$

Calculate the distance between each instance

$$\mathrm{First}(i,j) = \begin{cases} 1, & \text{if } a_i^1 = j \text{ or } a_j^1 = i \text{ or } a_i^1 = a_j^1 \\ 0, & \text{otherwise} \end{cases}$$

Divide groups based on the first neighbor

FABN

(1) Purpose

FABN combines TFN and SBN, leveraging source domain statistics to compensate for the lack of generic knowledge in TFN.

(2) Implementation

$$\mu_F^i = \frac{1}{bL}\sum_{b,L} F_{b;c;L}^i,\ \sigma_F^i = \sqrt{\frac{1}{bL}\sum_{b,L}\left(F_{b;c;L}^i - \mu_F^i\right)^2}.$$

Calculate the TFN for each group

$$\mu_{\mathrm{FABN}}^i = \alpha\mu_s + (1-\alpha)\mu_F^i,$$
$${\sigma_{\mathrm{FABN}}^i}^2 = \alpha\sigma_s^2 + (1-\alpha){\sigma_F^i}^2,$$

Integrate TFN and SBN

$$\mathrm{FABN}^i\left(\mu_{\mathrm{FABN}}^i, \sigma_{\mathrm{FABN}}^i\right) = \varphi\cdot\frac{\left(F_{;c;}^i - \mu_{\mathrm{FABN}}^i\right)}{\sqrt{{\sigma_{\mathrm{FABN}}^i}^2+\varepsilon}} + \beta,\quad 1\le i\le r,$$

S-FABN

(1) Purpose

S-FABN selects only the layers sensitive to domain shift for clustering, improving inference efficiency

(2) Implementation

$$\mathcal{KL}(\mathcal{D}_T \parallel \mathcal{D}_S) = \frac{\sigma_T^2 + (\mu_T - \mu_S)^2}{2\sigma_S^2} + \ln\left(\frac{\sigma_S}{\sigma_T}\right) - \frac{1}{2},$$

Calculate the KL divergence between the SBN and TBN

$$\text{Score} = (1 + \frac{1}{1 + \mathrm{e}^{-\mathcal{KL}_{st}}})\mathcal{KL}_{mean},$$

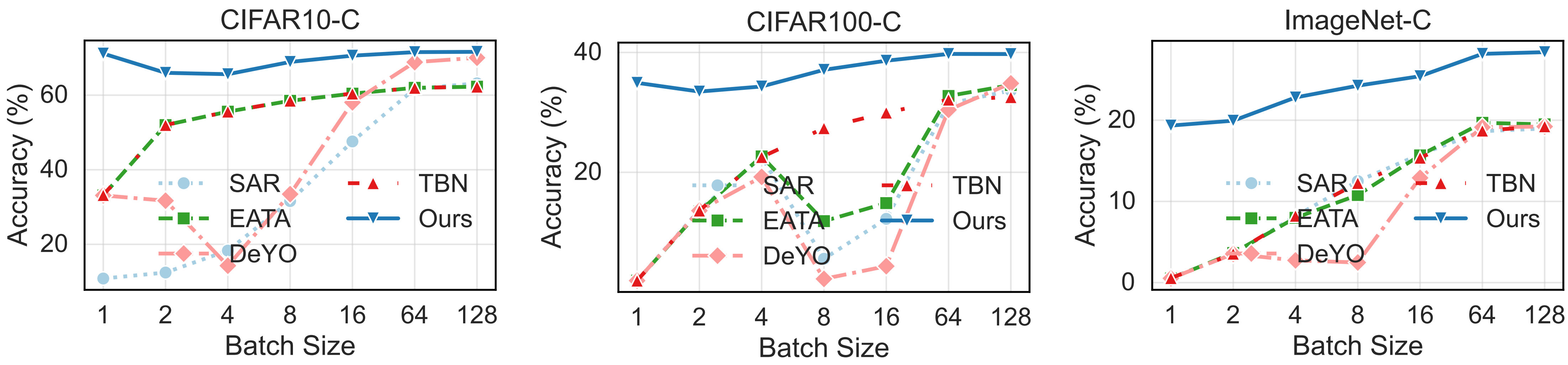
Calculate the score

Results

		CrossMix				Random				Shuffle				
Method	Venue	10-C	100-C	IN-C	Avg.	10-C	100-C	IN-C	Avg.	10-C	100-C	IN-C	Avg.	Avg-All
Source	CVPR16	57.39	28.59	25.64	37.21	57.38	28.58	25.93	37.30	57.38	28.58	25.80	37.25	37.25
TEST-TIME FINE-TUNE														
TENT	CVPR21	62.77	31.57	18.45	37.60	71.50	40.96	23.15	45.20	71.56	40.73	23.78	45.36	42.72
EATA	ICML22	61.97	32.74	19.68	38.13	68.25	42.15	24.26	44.89	67.83	41.37	24.50	44.57	42.53
NOTE	NIPS22	63.03	32.96	17.44	37.81	62.81	33.19	21.64	39.21	65.34	35.12	22.98	41.15	39.39
SAR	ICLR23	61.70	31.45	18.65	37.27	71.04	40.92	23.62	45.19	71.49	39.58	23.50	44.86	42.44
RoTTA	CVPR23	43.70	24.05	21.85	29.87	48.68	23.80	20.39	30.96	54.79	29.29	22.71	35.60	32.14
ViDA	ICLR24	61.97	32.14	18.52	37.54	67.96	39.48	23.33	43.59	67.96	39.48	23.06	43.50	41.54
DeYO	ICLR24	68.85	30.43	19.13	39.47	**75.63**	36.67	24.32	45.54	**75.62**	35.45	25.21	45.43	43.48
TEST-TIME NORMALIZATION														
TBN	ICML20	61.96	32.12	18.72	37.60	67.75	39.16	22.99	43.30	67.63	39.22	23.35	43.40	41.43
α-BN	arXiv20	62.41	33.22	21.92	39.18	69.88	41.59	27.57	46.35	69.87	41.60	27.78	46.42	43.98
IABN	NIPS22	62.63	24.54	9.73	32.30	64.59	26.40	10.75	33.91	64.59	26.40	10.79	33.93	33.38
FIND	Proposed	**71.54±0.2**	39.75±0.2	29.21±0.0	46.83	73.09±0.2	42.56±0.2	30.12±0.2	48.59	72.74±0.1	42.87±0.3	30.00±0.2	48.54	47.87
FIND*	Proposed	70.75±0.0	**40.48±0.1**	**30.33±0.1**	**47.19**	73.68±0.0	**43.05±0.0**	**30.62±0.0**	**49.12**	73.60±0.0	**43.88±0.2**	**30.24±0.4**	**49.23**	**48.50**

FIND significantly outperforms other TTA methods when dealing with dynamic data patterns.

Results



FIND demonstrates stable performance under different batch sizes.

Thank You!