AcTTA: Rethinking Test-Time Adaptation via Dynamic Activation

Supplementary Material

A. Implementation Details

All TTA experiments were implemented using a public benchmark repository on GitHub, https://github.com/mariodoebler/test-time-adaptation.

Optimizer and Learning Rate Settings. The optimizer and learning rate configurations used for each dataset and model are as follows: CIFAR-10 and CIFAR-100 experiments use the Adam optimizer with a learning rate of 1×10^{-3}. For ImageNet with ResNet-50, we use SGD with momentum 0.9 and a learning rate of 2.5×10^{-4} (applied to both BN and GN variants). For ImageNet with ViT models, we use SGD with momentum 0.9 and a learning rate of 1×10^{-3}. For the case of batch size 4, we scale the base learning rate by a factor of 0.1 for every method.

Dataset Settings. For the corrupted CIFAR datasets (CIFAR10-C and CIFAR100-C), we use all 10,000 test images. For ImageNet, we evaluate on 5,000 validation images following standard practice.

Hardware. All experiments were conducted using an NVIDIA RTX A6000 GPU and an AMD EPYC 7513 CPU.

B. Derivation of the AcTTA Activation

In this section, we formally derive the proposed AcTTA activation function. We begin by unifying modern activation functions under a common form, analyze their gradient dynamics, and extend them to a learnable, shift-aware formulation suitable for test-time adaptation.

B.1. Unified Representation of Base Activations

As discussed in prior work [1], most modern activation functions $\phi(x)$ can be represented as a self-gated function:

$$\phi(x) \approx x \cdot \sigma(\beta x), \tag{1}$$

where $\sigma(\cdot)$ denotes the sigmoid function and β is a scaling factor. This formulation unifies several representative nonlinearities:

- **ReLU**: Corresponds to $\beta \to \infty$ (in practice, $\beta > 10$).
- **Swish / SiLU**: Corresponds to $\beta = 1$.
- **GELU**: Can be approximated with $\beta \approx 1.702$.

B.2. Gradient Perspective of Activation Functions for Adaptation

Most modern activation functions can be interpreted from two complementary perspectives: a **gating perspective**, which determines which parts of the representation become active, and a **gradient perspective**, which captures how information flows backward under that gating pattern. In common piecewise activations such as ReLU, these two perspectives are tightly coupled. Neurons gated "off" suppress their forward signal and simultaneously receive zero gradient, whereas active neurons transmit both activation and gradient. In TTA, however, this coupling becomes particularly restrictive. Since TTA is itself an on-the-fly adaptation process, we posit that the ability to adjust representations in the negative (non-activated) region is crucial. If gradients vanish in this region, the model cannot exploit potentially informative directions for adaptation. Motivated by this insight, we aim to **control the gradient behavior** independently of the gating structure, enabling **non-zero and informative gradients even in nominally inactive regions** while retaining the beneficial gating property. This decoupling offers greater flexibility and supports more effective adaptation dynamics during TTA.

B.3. Generalizing the Gradient Behavior

Let us analyze the derivative of the base function $\phi(x)$:

$$\phi'(x) = \sigma(\beta x) + \beta x \cdot \sigma(\beta x)(1 - \sigma(\beta x)). \tag{2}$$

For practical values of $\beta \geq 1$, the gradient $\phi'(x)$ exhibits specific asymptotic behaviors:

$$\lim_{x \to -\infty} \phi'(x) = 0, \quad \lim_{x \to +\infty} \phi'(x) = 1. \tag{3}$$

This confirms that standard activations act as a fixed gate with restricted bounds: blocking gradients in the negative limit (0) and passing them with a unity scale in the positive limit (1).

To enable learnable adaptation, we require a gradient function whose slope is not restricted to the binary values $0, 1$ but instead can approach arbitrary optimal limits. Let the desired asymptotic slopes in the negative and positive regions be λ_{neg} and λ_{pos}, respectively. Formally, we seek a generalized slope function $\lambda(x)$ satisfying

$$\lim_{x \to -\infty} \lambda(x) = \lambda_{neg}, \quad \lim_{x \to +\infty} \lambda(x) = \lambda_{pos}. \tag{4}$$

A natural way to achieve this behavior is to start from a smooth gating function that transitions monotonically between two endpoints. The sigmoid $\sigma(\beta x)$ is particularly suitable because it smoothly interpolates from 0 (as $x \to -\infty$) to 1 (as $x \to +\infty$). If we interpret this sigmoid output as a continuous interpolation coefficient, then constructing a slope function that moves smoothly between λ_{neg} and λ_{pos} becomes straightforward.

Following this, the slope at input x is obtained by linearly blending the two endpoints according to the sigmoid gate:

$$\lambda(x) = \lambda_{neg} + (\lambda_{pos} - \lambda_{neg}) \cdot \sigma(\beta x). \tag{5}$$

Thus, the proposed formulation emerges directly from interpreting the sigmoid as a smooth interpolation mechanism. Rather than choosing $\lambda(x)$ arbitrarily, this construction provides the canonical way to transition between two learnable asymptotic slopes while preserving the smooth gating property.

B.4. Derivation via Integral Approximation

Our objective is to construct a new activation function $g(x)$ that satisfies two conditions:

1. **Backward Compatibility:** It should be able to express the pre-trained baseline $\phi(x)$.
2. **Gradient Matching:** The derivative of its residual term should approximate the target slope form $\lambda(x)$.

We formulate $g(x)$ as the base activation plus a learnable residual term:

$$g(x) = \phi(x) + \int \lambda(x)\, dx. \tag{6}$$

However, the exact integral of $\lambda(x)$ involves logarithmic terms (Softplus), which introduces additional computational overhead. Instead, we employ a simple approximation. Consider the term $x \cdot \lambda(x)$. Its derivative is:

$$\frac{d}{dx}[x \cdot \lambda(x)] = \lambda(x) + x \cdot \lambda'(x). \tag{7}$$

In the asymptotic regions ($x \to \pm\infty$), $\lambda(x)$ converges to a constant, causing $\lambda'(x) \to 0$. Consequently, the term $x \cdot \lambda'(x)$ vanishes, and the derivative approximates the target slope:

$$\frac{d}{dx}[x \cdot \lambda(x)] \approx \lambda(x). \tag{8}$$

Thus, we can efficiently approximate the integral term by simply adding $x \cdot \lambda(x)$.

B.5. Shift-Aware Formulation

Finally, to address the misalignment between the activation function and shifted feature distributions (often caused by normalization statistics), we introduce a learnable center parameter c. By substituting x with $(x - c)$, we allow the activation to operate around the target-domain mean.

Combining the shifted base activation with the approximated residual term yields the final AcTTA formulation:

$$g(x) = \phi(x - c) + \underbrace{\left[\lambda_{neg} + (\lambda_{pos} - \lambda_{neg})\sigma\big(\beta(x - c)\big)\right]}_{\lambda(x-c)} \cdot (x - c). \tag{9}$$

Our formulation also provides two important properties. First, it supports an **identity initialization**: by setting $\lambda_{neg} = \lambda_{pos} = 0$ and $c = 0$, the function reduces to $g(x) = \phi(x)$, ensuring that the model behaves identically to the original network at deployment time. Second, it offers clear **asymptotic gradient behavior**. As $x \to -\infty$, where $\phi'(x) \to 0$, the effective gradient of $g(x)$ converges to λ_{neg}; and as $x \to +\infty$, where $\phi'(x) \to 1$, the gradient approaches $1 + \lambda_{pos}$. This guarantees that the network can learn to exploit meaningful gradients in both negative and positive regions, offering dynamic and flexible adaptation behavior.

C. Comparison of Activation Parameterization Granularity

Method	Granularity	# param	Error (%)	
			BS=4	BS=128
TENT		17,952	30.36	18.51
AcTTA$_{\text{TENT}}$	channel	3,408	**29.56**	**17.03**
	layer	24	30.16	18.78
	pixel	3,489,792	48.58	58.80

Table 1. Ablation study on the degree of activation flexibility under different batch sizes on CIFAR10-C with a WRN-28 architecture. The best performance is obtained when activation flexibility is controlled on a per-channel basis.

We investigate how the granularity of activation-function parameterization influences adaptation performance, comparing channel-wise, layer-wise, and pixel-wise configurations. As shown in Table. 1, the model exhibits clear differences depending on how the activation flexibility is allocated. Channel-wise activation shows a notable advantage, achieving a lower error rate than both layer-wise and pixel-wise alternatives. This indicates that distributing activation flexibility across channels provides an effective balance between representational adaptability and parameter efficiency. In other words, channel-wise parameterization represents an intermediate point between the structural simplicity of layer-wise modulation and the high representational freedom of pixel-wise modulation, offering a balanced level of parameterization and expressiveness.

D. Additional Results on ImageNet-A, ImageNet-R, and PACS

We further evaluate AcTTA on ImageNet-A, ImageNet-R, and PACS to verify that its effectiveness is not limited to corruption benchmarks. As shown in Table. 2, incorporating AcTTA consistently improves three representative TTA baselines across all three datasets. These gains are observed not only on corruption-based benchmarks, but also under

more diverse and realistic distribution shifts, including natural adversarial examples in ImageNet-A, rendition shifts in ImageNet-R, and cross-domain shifts in PACS.

	TENT		ETA		DeYO	
Method	Base	AcTTA	Base	AcTTA	Base	AcTTA
ImageNet-A	66.13	**59.31**	63.52	**61.33**	61.27	**57.29**
ImageNet-R	48.96	**47.18**	44.55	**42.01**	41.68	**41.07**
PACS	16.75	**15.14**	18.20	**18.09**	16.58	**13.09**

Table 2. Additional results on ImageNet-A, ImageNet-R, and PACS.

These results suggest that the benefit of AcTTA does not depend on a specific corruption type or dataset structure. Rather, activation-function modulation appears to provide a generally useful adaptation mechanism that remains effective across a broad range of test-time distribution shifts. This consistency supports our view that AcTTA is not merely a corruption-specific design, but a more general approach for improving robustness through online activation adjustment during test-time adaptation.

E. Additional Experiments on Alternative Activations

We further compare AcTTA with alternative activation functions in Table. 3 and observe trends consistent with those reported in Table 3 of the main paper. Across both CIFAR10-C and CIFAR100-C, AcTTA consistently achieves the best performance, whereas alternative activation functions often provide limited gains or even cause substantial degradation under test-time adaptation. In particular, PReLU and PAU frequently exhibit severe instability, leading to much higher error rates than the standard baseline in several cases. We emphasize that such behavior is expected and reflects an inherent limitation of directly applying activation functions designed for standard training to the TTA setting.

	CIFAR10-C			CIFAR100-C		
Method	TENT	ETA	DEYO	TENT	ETA	DEYO
Base	18.55	18.33	18.44	31.68	31.24	30.41
PReLU	31.73	31.56	31.61	60.97	66.53	62.70
PAU	37.50	36.38	34.81	95.29	91.13	88.56
ACON	18.06	17.92	18.05	30.97	31.51	30.61
AcTTA	**17.11**	**17.00**	**17.29**	**30.64**	**30.99**	**30.03**

Table 3. Comparison with alternative activation functions. Error rate (%). Lower is better.

This can be understood from the mismatch between conventional training and TTA. These activation functions were originally developed for *training-time* optimization with abundant data and many update iterations, whereas TTA operates in a considerably more unstable online adaptation regime under distribution shift. As a result, activation designs that are effective during standard training do not necessarily remain reliable in TTA and may instead amplify optimization instability. Moreover, tuning these methods separately for each test distribution would contradict the practical objective of TTA. In addition, some alternatives such as PAU incur substantial computational overhead. Overall, these results suggest that naively transferring training-time activation designs to TTA exposes their inherent limitations, whereas AcTTA is better aligned with the TTA setting and remains robust.

F. Gradient Flow Analysis

We further analyze gradient flow by measuring the layer-wise gradient norms throughout the network. We also measure the fraction of gradients passing through the activation Figure. 1. Compared with TENT using ReLU, AcTTA exhibits consistently larger gradient norms and higher gradient pass-through rates. These observations suggest that AcTTA facilitates gradient propagation during test-time adaptation.

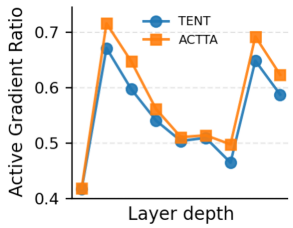

Figure 1. Gradient pass-through ratio.

Supplementary References

[1] Ningning Ma, Xiangyu Zhang, Ming Liu, and Jian Sun. Activate or not: Learning customized activation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8032–8042, 2021.