Purge-Gate: Backpropagation-Free Test-Time Adaptation for Point Clouds Classification via Token purging

Supplementary Material

6. Supplementary Material

6.1. Effect of Batch Size.

Figure 4 shows that increasing batch size improves accuracy for both methods, but PG-SP consistently outperforms TENT across all batch sizes. Unlike TENT, which struggles with small batches, PG-SP maintains high accuracy even in low-resource settings, demonstrating its robustness and adaptability. This highlights a key advantage of PG-SP: it performs well without relying on large batch sizes, making it more practical for real-world deployment with memory or computational constraints. This evaluation is performed on the ShapeNet-C dataset.

6.2. Real-world ScanObjectNN variants.

Table 5 confirms that our **PG-SP** strategy improves performance even under genuine domain shifts that are free of synthetic corruptions. Starting from the OBJ-ONLY model used in Table 1, we deploy PG-SP on two challenging variants: *OBJ-BG*, which introduces cluttered backgrounds, and *PB-T50-RS*, which combines point-based sampling and rotation. PG-SP raises accuracy from **74.18%** to **75.56%** on OBJ-BG (**+1.38%**), and from textbf56.77% to **61.07%** on PB-T50-RS (**+4.30%**). These gains demonstrate that our token-purging mechanism benefits real-world scenarios beyond the hand-crafted noise studied in ScanObjectNN-C.

Variant	Source Only	PG-SP (Ours)	Δ
OBJ-BG	74.18	**75.56**	+1.38
PB-T50-RS	56.77	**61.07**	+4.30

Table 5. Top-1 classification accuracy (%) on real-world ScanObjectNN variants without synthetic corruptions.

6.3. Dynamics of Purging and Entropy and its effect on the final ACC

Effect of purge size $\mathcal{L}_{\text{pg}}$ on ACC under *Distortion* corruption. Figure 7 presents Top-1 accuracy (black, left axis) and logit entropy (red, right axis) for 128 independent runs spanning $\mathcal{L}_{\text{pg}} \in [0, 127]$. Accuracy starts high at $\mathcal{L}_{\text{pg}} = 0$ (59 %), climbs slightly to a plateau of 63 % around $\mathcal{L}_{\text{pg}} \approx 48$, then drops sharply once purging exceeds 80, while entropy steadily rises and spikes when informative tokens are removed. The green dashed line indicates the maximum accuracy achieved by an exhaustive search restricted to $\mathcal{L}_{\text{pg}} \in [0, 32]$. The yellow band marks the six purge sizes $\{0, 2, 4, 8, 16, 32\}$ examined by our entropy-guided schedule; from this sparse set, it selects $\mathcal{L}^*_{\text{pg}}$ and attains 62.48 % accuracy (orange dashed line), essentially matching the global optimum without exhaustive tuning, thereby validating entropy as a practical proxy for near-optimal hyperparameter selection.

Dynamics of purging: interplay between accuracy and entropy (no $\mathcal{L}_{\text{pg}}$ selection). Figure 8 traces Top-1 accuracy (black, left axis) and logit entropy (red, right axis) for *Background* corruption level 3 as the purge size $\mathcal{L}_{\text{pg}}$ sweeps the full range $[0, 127]$; unlike earlier plots, the result of our entropy-guided schedule is *not* shown. Accuracy and entropy evolve inversely: starting at $\mathcal{L}_{\text{pg}} = 0$, accuracy is low (27 %) while entropy is high (0.60). Incrementally removing the most corrupted tokens ($0 < \mathcal{L}_{\text{pg}} \lesssim 64$) steadily boosts accuracy to a plateau of 64 % and drives entropy down to 0.25, indicating increased model confidence. Beyond $\mathcal{L}_{\text{pg}} \gtrsim 90$, further purging excises informative tokens; entropy rebounds and accuracy collapses, exposing the regime where pruning becomes detrimental. The green dashed line marks the best score achievable by an exhaustive yet *narrow* search over $\mathcal{L}_{\text{pg}} \in [0, 32]$, yielding only 38 %—well below the true optimum around $\mathcal{L}_{\text{pg}} \approx 64$. These dynamics underline the need for a principled, entropy-aware selection strategy: without it, practitioners risk settling on sub-optimal hyper-parameters or over-purging, both of which degrade performance.

Purging dynamics under *Background* corruption, severity 7 (no $\mathcal{L}_{\text{pg}}$ selection shown). Figure 9 plots Top-1 accuracy (black, left axis) and logit entropy (red, right axis) across 128 purge sizes $\mathcal{L}_{\text{pg}} \in [0, 127]$. At $\mathcal{L}_{\text{pg}} = 0$, heavy corruption yields low accuracy (13 %) and high entropy (0.83). Removing a handful of tokens ($0 < \mathcal{L}_{\text{pg}} \leq 32$, yellow band) barely helps—the best accuracy reachable in this restricted range (green dashed line) is only 16 %. As purging continues beyond $\mathcal{L}_{\text{pg}} \approx 40$, accuracy climbs almost linearly while entropy declines, peaking near $\mathcal{L}_{\text{pg}} \approx 112$ with 33 % accuracy and 0.45 entropy. Past this point the trends reverse: over-purging ($\mathcal{L}_{\text{pg}} > 120$) discards informative tokens, causing accuracy to collapse and entropy to rebound. These dynamics reveal that severe corruption demands far more aggressive purging than the conventional search window $[0, 32]$ can capture; without an adaptive, entropy-aware mechanism, practitioners risk selecting sub-optimal hyperparameters that leave most potential performance untapped.

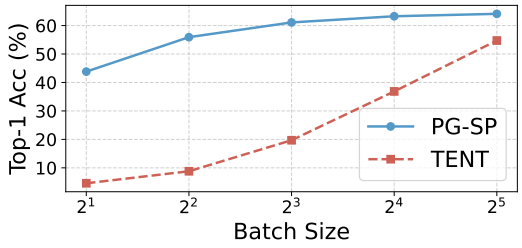

Figure 4. Effect of Batch size on ACC performance of our PG-SP and TENT, on ShapeNet-C dataset.

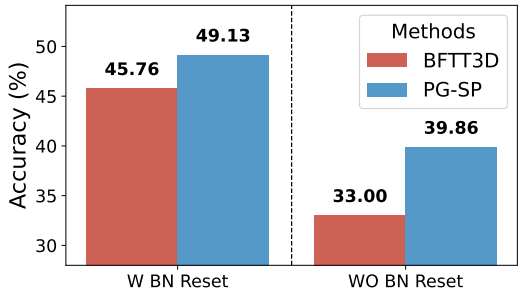

Figure 5. Effect of BatchNorm reset on our PG-SP, and our baseline BFTT3D, on the ScanObjectNN-C dataset. Values are the mean of ACC

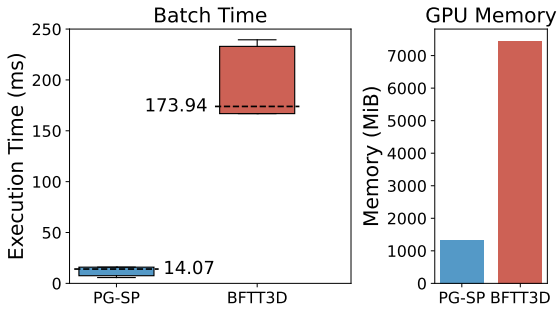

Figure 6. Comparation of GPU time and memory for our PG-SP, and our baseline BFTT3D method. Both are evaluated with a batch size of 32 on ModelNet-C dataset.

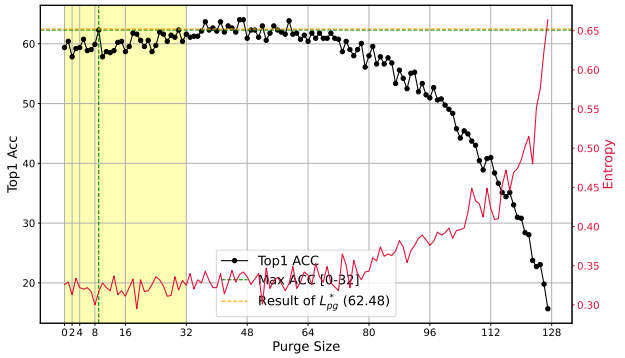

Figure 7. Effect of purge size on ACC of **Distortion** corruption from ScanObjectNN-C dataset.

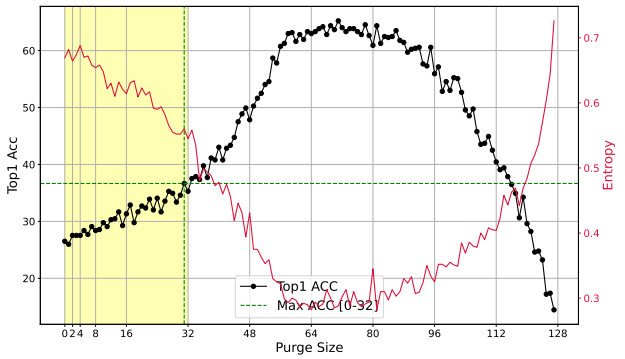

Figure 8. Effect of purge size on ACC of **Background** corruption **level 3** from ScanObjectNN-C dataset.

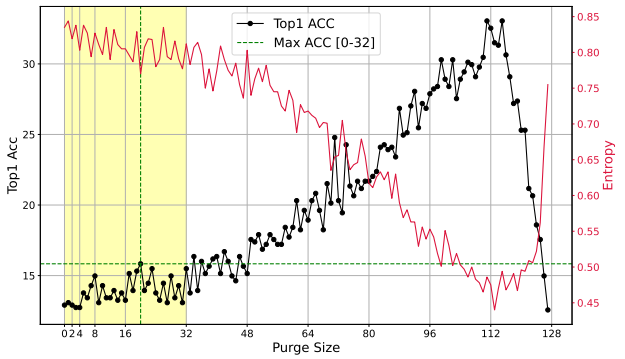

Figure 9. Effect of purge size on ACC of **Background** corruption **level 7** from ScanObjectNN-C dataset.