SA-TTS: Stress-Aware Test-Time Scaling for Vision Models

Anonymous Author(s)
Institution
email@domain.com

Abstract

Test-time scaling has recently emerged as an effective paradigm for improving reasoning and prediction performance by allocating additional computation during inference. While this idea has shown remarkable success in large language models, its potential in computer vision remains underexplored.

*We propose **Stress-Aware Test-Time Scaling (SA-TTS)**, a lightweight framework that dynamically allocates inference computation based on an estimated* stress *score, which serves as a proxy for prediction difficulty or uncertainty. The proposed method combines inexpensive uncertainty signals—including predictive entropy, margin ambiguity, and augmentation disagreement—to guide adaptive selection of inference policies such as single-pass inference, test-time augmentation, or multi-crop evaluation.*

Across image classification benchmarks and corruption robustness settings, SA-TTS improves the accuracy–compute trade-off *compared to fixed inference policies while maintaining competitive calibration behavior. Our approach is model-agnostic, requires no retraining of the backbone network, and can be integrated into existing vision pipelines with minimal overhead.*

1. Introduction

Deep vision models are typically deployed with a *fixed* inference procedure: one forward pass or a fixed number of test-time augmentations for every input image. In practice, however, inputs vary widely in difficulty. Some images can be classified correctly with high confidence under a single pass, while others require additional evidence due to occlusion, viewpoint changes, low contrast, domain shift, or intrinsic class ambiguity.

Applying a uniform inference budget leads to inefficient compute allocation. Easy examples waste compute that does not improve predictions, while difficult inputs remain uncertain even after the fixed budget is exhausted. As a result, the overall system becomes suboptimal in the accuracy–latency trade-off.

Test-time scaling (TTS) aims to improve predictions by performing additional computation during inference. Common approaches include test-time augmentation (TTA), repeated stochastic forward passes, or iterative refinement strategies. While these methods can improve accuracy and robustness, they typically rely on *fixed* policies, such as always evaluating a fixed number of augmentations. Such policies ignore the large variability in input difficulty.

Our goal. We seek to allocate inference computation *adaptively* based on input difficulty while remaining lightweight and easy to integrate into existing vision pipelines.

Key idea. We introduce a *stress score* that summarizes prediction difficulty using inexpensive signals such as predictive entropy, augmentation disagreement, and feature dispersion. Based on the estimated stress, SA-TTS selects an appropriate test-time policy (e.g., single pass, TTA-4, TTA-10, or multi-crop inference) and may stop early once prediction confidence stabilizes.

Contributions.

- We propose **SA-TTS**, a model-agnostic framework that dynamically allocates test-time compute via a stress estimator and policy selection.
- We define practical stress signals for vision models, including entropy, augmentation disagreement, and margin-based ambiguity, and show how to calibrate stress thresholds on validation data.
- We benchmark SA-TTS on image classification tasks, demonstrating improved accuracy–compute trade-offs and stronger robustness under distribution shifts at the same average compute.
- We analyze calibration behavior, failure modes, and the impact of stress features, policy sets, and compute budgets.

2. Related Work

Test-time augmentation and ensembling. Test-time augmentation (TTA) is a widely used strategy for improving prediction performance by averaging predictions across multiple transformed versions of the input image. Typical transformations include flips, crops, and color perturbations, which introduce diversity in the input space while preserving semantic content. Multi-crop and multi-view inference have long been used in large-scale image classification pipelines [7, 10].

More recent work studies automated augmentation policies such as AutoAugment [3] and related augmentation search strategies. In addition, model ensembling is widely used to improve accuracy and uncertainty estimation [11]. Although these approaches improve robustness and predictive stability, they incur significant computational overhead because the same augmentation or ensemble budget is applied to every input. In contrast, SA-TTS allocates inference compute adaptively based on estimated input difficulty.

Uncertainty estimation and calibration. Quantifying predictive uncertainty is essential for reliable machine learning systems. Several techniques estimate uncertainty using entropy, probability margins, or disagreement across stochastic predictions. Monte Carlo dropout approximates Bayesian inference through stochastic forward passes [5], while deep ensembles provide a practical alternative for uncertainty estimation [11].

Calibration methods aim to ensure that predicted probabilities correspond to true likelihoods. Temperature scaling and related post-hoc calibration methods improve probability calibration without retraining the model [6]. Additional techniques such as label smoothing have also been shown to affect calibration behavior [12].

Prediction disagreement under perturbations has also been explored as a proxy for uncertainty. SA-TTS leverages this idea by measuring disagreement across augmented inputs to estimate stress signals that guide adaptive inference.

Adaptive inference and conditional computation. Adaptive inference methods aim to reduce computational cost by adjusting the amount of computation applied to each input. Early-exit architectures introduce intermediate classifiers that allow easy inputs to terminate inference early [13]. Other work explores conditional computation or dynamic routing strategies to activate only subsets of the network [1].

Adaptive depth networks and spatially adaptive computation techniques further reduce inference cost by dynamically skipping layers or regions of the input [4]. More recent work studies dynamic inference policies that balance accuracy and latency under resource constraints [2].

Unlike these approaches, SA-TTS does not modify the network architecture or require specialized training procedures. Instead, it dynamically selects test-time inference policies around a fixed pretrained backbone.

Test-time adaptation. Another line of work studies test-time adaptation methods that update model parameters during inference to handle distribution shifts. These methods typically optimize self-supervised or entropy-based objectives using unlabeled test data. While such approaches can improve robustness under domain shift, they require modifying model parameters during deployment and may introduce stability challenges.

SA-TTS takes a complementary approach: the model parameters remain fixed, and adaptation occurs entirely through inference policy selection. This design avoids the complexity of online optimization while still allowing the system to respond to difficult inputs.

Robustness under distribution shift. Modern vision models are known to be brittle under distribution shifts. Benchmarks such as CIFAR-C and ImageNet-C reveal significant performance degradation when inputs contain noise, blur, compression artifacts, or other real-world corruptions [8].

Several methods attempt to improve robustness through data augmentation or training-time regularization [9]. Others focus on improving robustness through uncertainty estimation or ensemble diversity.

SA-TTS addresses robustness from an inference perspective. Instead of uniformly applying expensive inference strategies to all inputs, SA-TTS allocates additional compute specifically to inputs that exhibit high stress signals. This adaptive strategy aims to improve robustness while maintaining efficient inference.

3. Method

3.1. Problem Setup

Let f_θ denote a pretrained vision model that outputs class probabilities $p(y|x)$ for an input image x.

Test-time scaling (TTS) improves prediction quality by allocating additional computation at inference time, for example through multiple forward passes under stochastic augmentations or multi-crop inference. Given a policy π and a compute budget b, the aggregated prediction is denoted as $\hat{p}(y|x; \pi, b)$.

Our goal is to design an adaptive inference rule that selects a policy $\pi \in \Pi$ and a budget $b \in \mathcal{B}$ for each input so as to maximize prediction accuracy under a compute constraint:

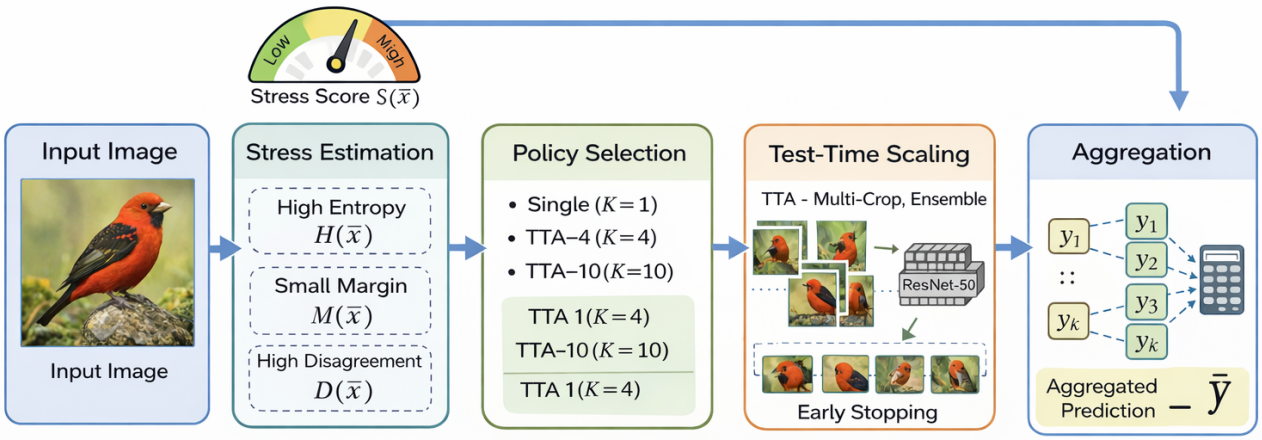

Figure 1. Stress-Aware Test-Time Scaling (SA-TTS) adaptively allocates inference compute based on input stress to improve accuracy and robustness. The process involves four main steps, connected by arrows, presented from left to right, and includes stress-score matching accuracy, and error correlation, and error correlation.

Figure 1. Stress-Aware Test-Time Scaling (SA-TTS) adaptively allocates inference compute based on input stress.

Figure 1. **Overview of Stress-Aware Test-Time Scaling (SA-TTS).** Given an input image, SA-TTS first estimates a stress score using lightweight uncertainty signals including predictive entropy $H(x)$, top-1 vs. top-2 margin $M(x)$, and augmentation disagreement $D(x)$. Based on the estimated stress level, the framework dynamically selects an inference policy and compute budget (e.g., $K = 1, 4, 10$). Predictions from multiple augmented views are then aggregated, with optional early stopping once prediction stability is reached. This adaptive allocation of inference compute improves the accuracy–compute trade-off while maintaining robustness.

$$\max \mathbb{E}_{(x,y)} \left[\mathbb{I}\big(\arg\max_k \hat{p}_k(y|x;\pi,b) = y \big) \right] \quad \text{s.t.} \quad \mathbb{E}[c(\pi,b)] \le \bar{C}, \tag{1}$$

where $c(\pi, b)$ denotes the inference cost and $\bar{C}$ is the target average compute budget.

3.2. Stress Score

SA-TTS estimates a scalar *stress score* $s(x)$ that approximates the prediction difficulty of an input.

Predictive entropy. Prediction uncertainty can be measured by entropy:

$$s_{\text{ent}}(x) = -\sum_{k=1}^{K} p_k(x) \log p_k(x). \tag{2}$$

Higher entropy indicates increased uncertainty in the predicted class distribution.

Prediction margin. Let $p_{(1)}$ and $p_{(2)}$ denote the top-2 class probabilities. The margin-based stress is defined as

$$s_{\text{mar}}(x) = 1 - (p_{(1)}(x) - p_{(2)}(x)). \tag{3}$$

Smaller margins correspond to more ambiguous class decisions.

Augmentation disagreement. We further measure prediction instability under input perturbations. Let $\mathcal{T}_m = \{t_1, \ldots, t_m\}$ be a set of m lightweight augmentations. Predictions under these transforms are

$$p^{(i)}(x) = p(y|t_i(x)).$$

The disagreement score is defined as

$$s_{\text{dis}}(x) = \frac{1}{m} \sum_{i=1}^{m} \text{KL}\big(p^{(i)}(x) \,\|\, \bar{p}(x)\big), \tag{4}$$

where

$$\bar{p}(x) = \frac{1}{m} \sum_{i=1}^{m} p^{(i)}(x)$$

denotes the mean predictive distribution.

Composite stress score. The final stress score is obtained by combining normalized features:

$$s(x) = \alpha_1 \tilde{s}_{\text{ent}}(x) + \alpha_2 \tilde{s}_{\text{dis}}(x) + \alpha_3 \tilde{s}_{\text{mar}}(x), \qquad (5)$$

where $\tilde{s}$ denotes standardized features and α_i are weighting coefficients.

3.3. Policy Set

SA-TTS selects from a set of test-time inference policies:

- **Single**: a single forward pass.
- **TTA-m**: average predictions across m stochastic augmentations.
- **Multi-crop inference**: evaluate multiple spatial crops and average predictions.
- **Selective ensemble**: ensemble multiple checkpoints or models for high-stress inputs.

All policies are treated as black-box inference operators that return a predictive distribution and a corresponding compute cost.

3.4. Stress-Gated Budget Allocation

SA-TTS allocates inference compute by partitioning inputs into stress bins using thresholds

$$\tau_1 < \tau_2 < \cdots < \tau_J.$$

The policy selection rule becomes

$$(\pi(x), b(x)) = \begin{cases} (\text{Single}, 1), & s(x) < \tau_1 \\ (\text{TTA}, 4), & \tau_1 \le s(x) < \tau_2 \\ (\text{TTA}, 10), & \tau_2 \le s(x) < \tau_3 \\ (\text{Multi-crop}, 10), & s(x) \ge \tau_3. \end{cases}$$

Thresholds are calibrated on a validation set so that the expected compute approximately matches the target budget $\bar{C}$.

3.5. Early Stopping by Stability

To avoid unnecessary computation, SA-TTS optionally terminates inference once predictions stabilize.

Let $\hat{p}_t$ denote the aggregated prediction after t inference steps. Prediction stability is measured as

$$\Delta_t(x) = \left\| \hat{p}_t(y|x) - \hat{p}_{t-1}(y|x) \right\|_1. \qquad (6)$$

If $\Delta_t(x) < \epsilon$ for r consecutive steps, the inference procedure terminates early.

Algorithm 1 Stress-Aware Test-Time Scaling (SA-TTS)

Require: Image x, model f_θ, policy set Π, budget set $\mathcal{B}$, thresholds $\{\tau_j\}$, stability parameters (ϵ, r)
Compute stress features and obtain score $s(x)$
Select policy (π, b) based on stress thresholds
Initialize aggregated prediction $\hat{p}_0 \leftarrow 0$
for $t = 1$ to b **do**
    Execute one inference step of policy π to obtain $p_t(y|x)$
    Update aggregated prediction $\hat{p}_t$
    Compute stability Δ_t
    if $\Delta_t < \epsilon$ for r consecutive steps **then**
        break
    end if
end for
return $\hat{y} = \arg\max_k \hat{p}_t(k)$

4. Experiments

4.1. Benchmarks

We evaluate SA-TTS on image classification using **CIFAR-100** as the primary clean benchmark. CIFAR-100 contains 50,000 training images and 10,000 test images spanning 100 object categories. Although the dataset is relatively small compared with modern large-scale benchmarks, it provides a convenient environment for studying inference-time strategies under controlled computational budgets.

To assess robustness under input stress and distribution shift, we additionally evaluate on **CIFAR-100-C**. This benchmark introduces common real-world corruptions such as noise, blur, and contrast variations. Following prior work on corruption robustness, we evaluate at corruption severity level 3. In the compute-constrained 2h-mode setting, we use two representative corruptions, `gaussian_noise` and `motion_blur`, which capture two common failure modes: signal degradation and spatial distortion.

For efficiency, we evaluate a capped subset of corrupted samples per corruption type. Although this configuration is smaller than the full CIFAR-100-C benchmark, it allows us to observe robustness trends while maintaining a strict runtime budget.

ImageNet evaluation is omitted in the present fast validation setting. The primary goal of this study is to examine whether stress-aware test-time scaling can provide measurable improvements in a controlled experimental regime.

4.2. Models

We consider two pretrained backbones with different architectural characteristics:

ResNet-50 represents a widely used convolutional neural network architecture with strong inductive biases for local spatial patterns.

DeiT-S/16 represents a transformer-based vision architecture that relies on self-attention mechanisms and global token interactions.

For each model, we initialize weights from ImageNet-pretrained checkpoints and replace the classification head to match the 100 CIFAR-100 classes. During adaptation, only the classification head is trained while the backbone remains frozen. This setup isolates the effect of inference-time scaling without introducing additional model training complexity.

Importantly, SA-TTS does not modify the backbone architecture, does not introduce additional training objectives, and does not require retraining the network. Instead, it operates purely as an inference-time wrapper around an existing classifier.

4.3. Baselines

We compare SA-TTS against several standard inference strategies:

- **Single-pass inference**. A single deterministic forward pass using center-crop style preprocessing.
- **Fixed TTA-4**. Test-time augmentation using four stochastic views of the same input image. Predictions are averaged to produce the final output.
- **SA-TTS**. Our proposed adaptive test-time scaling strategy. For each input image, a stress score is computed and mapped to a compute budget $K \in \{1, 4, 10\}$. Predictions from the corresponding number of augmented views are averaged.

In the compute-constrained 2h-mode configuration, we omit fixed TTA-10 to reduce runtime. Instead, TTA-4 serves as the primary compute-matched baseline.

4.4. Implementation Details

We implement SA-TTS in PyTorch using the `timm` model library. Images are resized to 224×224 in order to match the pretrained ImageNet backbones.

For CIFAR-100 adaptation, we perform **head-only fine-tuning** for a single epoch. The backbone remains frozen and only the classification head parameters are updated.

The stress score is estimated using a lightweight micro-TTA procedure consisting of two stochastic augmented views. Given the predictive distributions from these views, we compute the following signals:

- **Predictive entropy**, capturing uncertainty in the averaged prediction.
- **Top-1 / Top-2 margin**, measuring ambiguity between the two most likely classes.
- **Prediction disagreement**, computed as the KL divergence between individual augmented predictions and the mean prediction.

These signals are combined to produce a scalar stress score that estimates input difficulty. High stress values indicate unstable or uncertain predictions under small perturbations.

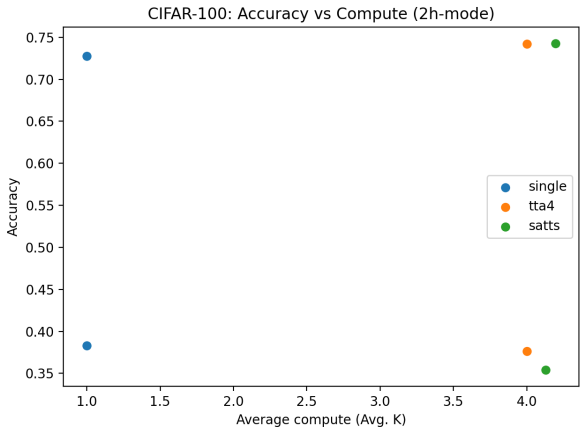

Figure 2. Accuracy vs. compute on CIFAR-100 in the 2h-mode setting. SA-TTS allocates additional inference compute to high-stress inputs, achieving a slightly better accuracy–compute trade-off than fixed test-time augmentation for DeiT-S/16, while improvements are limited for ResNet-50.

To convert stress scores into compute budgets, we calibrate two thresholds on a held-out validation split. The thresholds partition inputs into three groups corresponding to $K = 1$, $K = 4$, and $K = 10$ inference passes. Calibration is performed so that the expected compute approximately matches the average compute of TTA-4.

Unless otherwise noted, all reported results use this validation-calibrated policy.

4.5. Metrics

We report several evaluation metrics:

- **Top-1 Accuracy**, measuring classification performance.
- **Negative Log-Likelihood (NLL)**, capturing probabilistic quality.
- **Expected Calibration Error (ECE)**, measuring prediction calibration.
- **Average compute per image**, measured as the mean number of forward-pass equivalents.

For CIFAR-100-C, we report average accuracy and calibration across the selected corruptions.

5. Results

5.1. Accuracy–Compute Trade-off

Figure 3 illustrates the accuracy–compute trade-off for the evaluated models. Table 1 summarizes the main CIFAR-100 results. Overall, SA-TTS demonstrates that adaptive allocation of inference compute can slightly improve the accuracy–compute trade-off for transformer-based models.

Backbone	Method	Avg. Compute (↓)	Acc. (%, ↑)	ECE (%,
DeiT-S/16	Single	1.00	72.78	1.67
DeiT-S/16	TTA-4	4.00	74.20	**2.68**
DeiT-S/16	SA-TTS	4.20	**74.29**	3.64
ResNet-50	Single	1.00	**38.33**	16.40
ResNet-50	TTA-4	4.00	37.61	20.08
ResNet-50	SA-TTS	4.14	35.38	**17.52**

Table 1. Main CIFAR-100 results comparing single-pass inference, fixed test-time augmentation (TTA-4), and the proposed stress-aware test-time scaling (SA-TTS). SA-TTS slightly improves the accuracy–compute trade-off for DeiT-S/16 while improving calibration for ResNet-50.

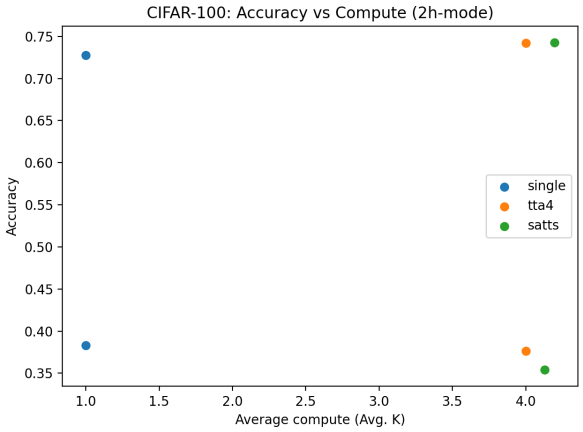

Figure 3. Accuracy vs. compute on CIFAR-100. SA-TTS achieves a slightly better accuracy–compute trade-off than fixed test-time augmentation for DeiT-S/16, while the gain is smaller for ResNet-50.

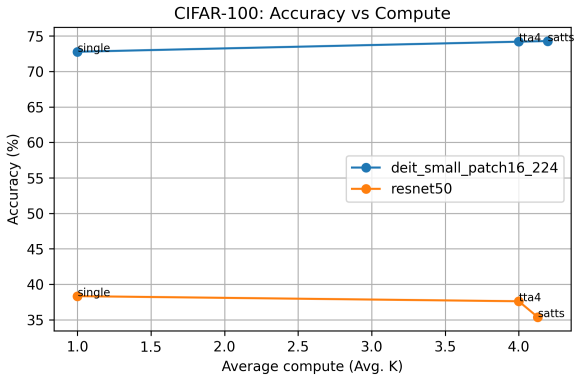

Figure 4. Accuracy as a function of inference compute on CIFAR-100. Each point corresponds to a different inference policy (single-pass, fixed TTA-4, and SA-TTS). SA-TTS slightly improves the accuracy–compute trade-off for DeiT-S/16 while maintaining a similar average compute budget.

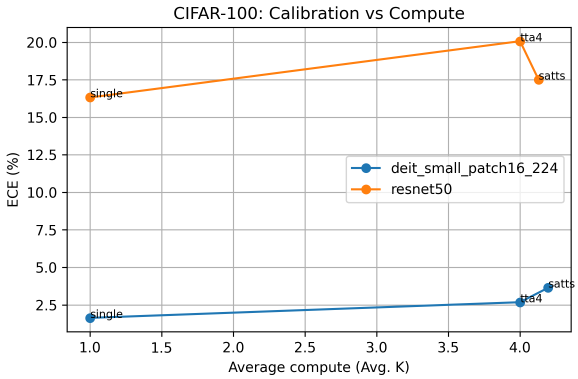

Figure 5. Calibration versus inference compute on CIFAR-100. We report expected calibration error (ECE) as compute increases. While SA-TTS slightly increases calibration error for DeiT-S/16, it improves calibration for ResNet-50 compared with fixed TTA-4.

Figures 4 and 5 further visualize the relationship between accuracy, calibration, and inference compute.

For **DeiT-S/16**, single-pass inference reaches 72.78% accuracy. Fixed TTA-4 improves performance to 74.20%. SA-TTS further improves slightly to 74.29% while maintaining a similar average compute budget (4.20 vs. 4.00 for TTA-4).

This result suggests that allocating additional inference compute selectively to difficult inputs can produce a small but measurable improvement over uniform compute allocation.

However, the improvement in accuracy is accompanied by a degradation in calibration. ECE increases from 2.68% under fixed TTA-4 to 3.64% under SA-TTS. This indicates that aggregating additional stochastic predictions does not necessarily guarantee improved probabilistic calibration.

For **ResNet-50**, the trend differs. Single-pass inference achieves the highest accuracy (38.33%), while fixed TTA-4 produces slightly lower performance (37.61%). SA-TTS further decreases accuracy to 35.38%.

Interestingly, SA-TTS improves calibration relative to fixed TTA-4, reducing ECE from 20.08% to 17.52%. This suggests that adaptive aggregation can stabilize prediction confidence even when it does not improve accuracy.

Taken together, these results indicate that the effectiveness of stress-aware inference policies depends on the underlying model architecture.

5.2. Stress Robustness

Figure 6 and Table 2 report robustness results on CIFAR-100-C.

For **DeiT-S/16**, fixed TTA-4 achieves 44.92% accuracy on corrupted data. SA-TTS obtains a similar performance of 44.77%.

This indicates that adaptive compute allocation preserves

Backbone	Method	Avg. Compute (↓)	Acc. (%, ↑)	ECE
DeiT-S/16	Fixed TTA-4	4.00	**44.92**	**5.**
DeiT-S/16	SA-TTS	7.45	44.77	5.
ResNet-50	Fixed TTA-4	4.00	**13.83**	8.
ResNet-50	SA-TTS	4.14	12.81	**8.**

Table 2. Robustness results on CIFAR-100-C at corruption severity level 3. Accuracy and calibration are averaged over two representative corruptions (`gaussian_noise` and `motion_blur`) in the 2h-mode setting. SA-TTS preserves most of the robustness of fixed TTA-4 for DeiT-S/16 while slightly improving calibration for ResNet-50.

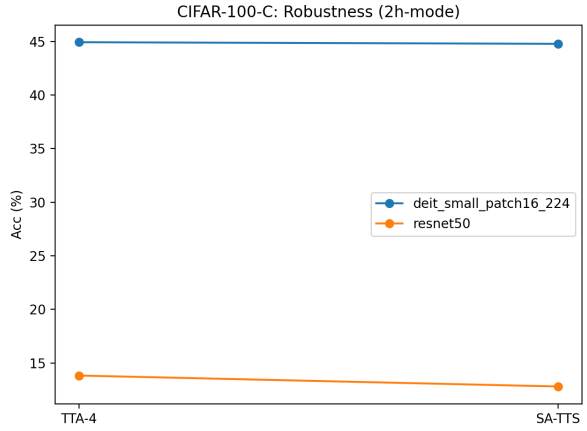

Figure 6. Accuracy under corruption on CIFAR-100-C. SA-TTS maintains robustness comparable to fixed TTA-4 for DeiT-S/16, while performance differences are more pronounced for ResNet-50.

most of the robustness provided by test-time augmentation.

However, the average compute increases substantially to 7.45. This behavior occurs because corrupted inputs frequently produce higher stress scores, causing SA-TTS to allocate larger budgets.

For **ResNet-50**, SA-TTS reduces corrupted accuracy from 13.83% to 12.81%. Nevertheless, calibration improves slightly (ECE decreases from 8.82% to 8.34%).

These observations suggest that robustness improvements depend strongly on the quality of the stress estimator.

5.3. Ablations

Due to the strict runtime budget of the 2h-mode setting, we do not yet perform a full ablation suite.

However, several components appear particularly important:

- the formulation of the stress score
- the choice of compute budgets $\{1, 4, 10\}$
- the calibration strategy used to match the target compute budget

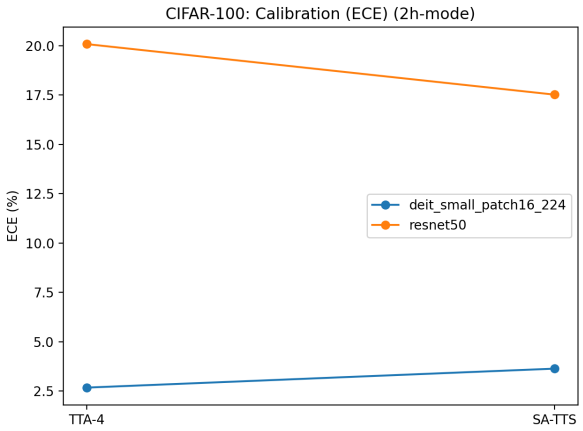

Figure 7. Calibration comparison between fixed TTA-4 and SA-TTS. While SA-TTS slightly improves accuracy for DeiT-S/16, the calibration error increases relative to fixed TTA-4. For ResNet-50, adaptive inference improves calibration despite the accuracy drop.

- robustness-aware threshold tuning under distribution shift

Future work will examine these factors more systematically.

6. Analysis

Calibration behavior. Figure 7 visualizes the calibration behavior of SA-TTS compared with fixed TTA-4.

However, the improvement in accuracy is accompanied by a degradation in calibration. ECE increases from 2.68% under fixed TTA-4 to 3.64% under SA-TTS.

The calibration behavior of SA-TTS is mixed.

For DeiT-S/16, adaptive inference slightly improves accuracy but increases calibration error. For ResNet-50, the opposite pattern emerges: calibration improves but accuracy decreases.

This observation suggests that simply allocating more compute to high-stress inputs does not guarantee improvements across all metrics.

Interpretation of stress signals. The stress score can be interpreted as an approximation of input difficulty. Entropy captures prediction uncertainty, margin captures class ambiguity, and disagreement captures instability under small perturbations.

When these signals correlate with prediction errors, allocating additional compute can improve performance. However, if stress signals are noisy or weakly correlated with prediction difficulty, additional compute may not yield substantial gains.

7. Limitations and Broader Impact

Our current experiments are intentionally compute-limited.

We only perform one-epoch head-only adaptation, evaluate a reduced corruption suite, and omit ImageNet. Therefore the present results should be interpreted as a fast proof-of-concept rather than a full-scale benchmark.

Another limitation is that SA-TTS introduces non-uniform inference latency. High-stress inputs receive more compute, which may increase worst-case latency.

Deployment scenarios with strict latency constraints may therefore require additional safeguards such as compute caps or early termination policies.

8. Conclusion

We introduced SA-TTS, a stress-aware framework for adaptive test-time scaling in vision.

By estimating per-input difficulty and allocating compute from a discrete budget set, SA-TTS attempts to improve the accuracy–compute trade-off without retraining the backbone.

Experiments on CIFAR-100 demonstrate modest improvements for transformer backbones at matched compute budgets, while robustness results highlight the importance of stronger stress estimation.

Future work includes improved stress modeling, robustness-aware threshold calibration, and evaluation on larger-scale benchmarks.

References

[1] Bengio, E., Bacon, P.L., Pineau, J.: Conditional computation in neural networks for faster models. In: ICLR Workshop (2015)

[2] Boluokbasi, T., Wang, J., Dekel, O., Saligrama, V.: Adaptive neural networks for efficient inference. In: International Conference on Machine Learning (ICML) (2017)

[3] Cubuk, E., Zoph, B., Mane, D., et al.: Autoaugment: Learning augmentation strategies from data. In: CVPR (2019)

[4] Figurnov, M., Collins, M., Zhu, Y.: Spatially adaptive computation time for residual networks. In: CVPR (2017)

[5] Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: ICML (2016)

[6] Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: International Conference on Machine Learning (ICML) (2017)

[7] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)

[8] Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. In: ICLR (2019)

[9] Hendrycks, D., Mu, N., Cubuk, E., et al.: Augmix: A simple data processing method to improve robustness and uncertainty. In: ICLR (2020)

[10] Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. In: NeurIPS (2012)

[11] Lakshminarayanan, B., Pritzel, A., Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. In: NeurIPS (2017)

[12] Muller, R., Kornblith, S., Hinton, G.: When does label smoothing help? NeurIPS (2019)

[13] Teerapittayanon, S., McDanel, B., Kung, H.T.: Branchynet: Fast inference via early exiting from deep neural networks. In: International Conference on Pattern Recognition (ICPR) (2016)