FRAGILITY-AWARE NOISE TRAINING FOR ROBUST SPLIT COMPUTING OVER NOISY CHANNELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Split computing (SC) partitions deep neural networks (DNNs) between edge and cloud devices, reducing edge-side computation while leveraging powerful cloud resources. However, in practice, intermediate features must traverse non-ideal communication links, where diverse perturbations can corrupt task-relevant representations and substantially degrade inference performance. To address this challenge, we develop a biologically inspired SC framework that progresses hierarchically from peripheral perception through neural transmission to central cognition, and further propose Fragility-Aware Noise Training (FNT). FNT estimates a relative fragility field over intermediate features, converts it into a bounded local SNR field, and allocates stronger but controlled perturbations to fragile locations under a min–max objective, encouraging the split model to learn more robust intermediate representations. Extensive experiments across diverse tasks, architectures, and split points show that FNT consistently improves robustness, achieving an average accuracy gain of 8.75% over the strongest baseline and a gain of 36.27% under the strongest additive perturbations. Moreover, under cross-noise evaluation, FNT retains strong cross-noise robustness, improving average accuracy by 10.75% and the performance under the strongest unseen perturbations by 28.19%. These results demonstrate that FNT improves the robustness of intermediate-feature transmission under diverse and unmatched channel perturbations.

1 INTRODUCTION

As deep neural networks (DNNs) become increasingly capable, there is growing demand for deploying intelligent inference on resource-constrained edge devices (Lin et al., 2020). However, the computational and memory demands of modern DNNs often make fully on-device inference impractical (Eshratifar et al., 2019). Cloud-only inference alleviates this burden but requires transmitting raw inputs, introducing additional communication overhead, latency, and energy consumption. Split Computing (SC) offers a practical compromise by partitioning a DNN between the edge and the cloud: the edge subnetwork extracts intermediate features, which are transmitted to the cloud for the remaining inference (Hu et al., 2019). Existing partitioning strategies aim to select suitable split points that balance edge computation and communication cost (Laskaridis et al., 2020). By jointly leveraging edge and cloud resources, SC enables efficient DNN inference with reduced edge-side workload and access to powerful cloud computation (Matsubara et al., 2022).

However, in practical SC systems, intermediate features must be transmitted from the edge to the cloud over real communication links, where they are inevitably exposed to channel noise, transmission errors, and feature loss (Shao & Zhang, 2020). Since these features encode task-relevant representations, such perturbations can propagate through the cloud-side network and degrade inference performance (Shao et al., 2021; Li et al., 2024). This makes conventional split computing frameworks that assume ideal feature transmission insufficient for capturing the coupling between computation and communication in realistic environments.

Inspired by the biological nervous system, we introduce a bio-inspired split computing framework based on a "peripheral perception–neural transmission–central cognition" pipeline (Fornos et al., 2019), as illustrated in Figure 1. In biological systems, neural signals are also subject to stochastic perturbations during transmission (Faisal et al., 2008; Ribrault et al., 2011). Accordingly, we model edge-side feature extraction, noisy intermediate-feature transmission, and cloud-side seman-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

tic processing as peripheral perception, neural transmission, and central cognition, respectively. This framework explicitly incorporates non-ideal communication into end-to-end inference, providing a unified perspective for studying intermediate-feature fragility and transmission robustness.

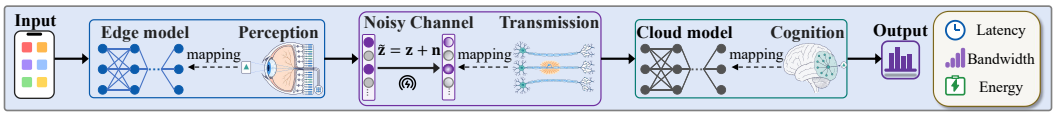


Figure 1: **Overview of the proposed bio-inspired SC framework.** It follows a peripheral perception–neural transmission–central cognition pipeline for noisy edge-to-cloud feature transmission.

Recent work on robust deep learning has explored random perturbation augmentation (Hendrycks et al., 2019), adversarial and worst-case training (Madry et al., 2017; Chen et al., 2026), and certified methods such as randomized smoothing (Cohen et al., 2019). However, many approaches focus on input-space or global representation robustness, with limited attention to intermediate feature transmission in split computing (Matsubara et al., 2022; Shao & Zhang, 2020). They also often assume predefined noise models or perturbation budgets, while practical communication conditions may vary in noise type, strength, and location (Xie et al., 2019). Moreover, intermediate features exhibit non-uniform task relevance across spatial positions and channels, making equal-magnitude perturbations affect predictions differently (Duan et al., 2023).

To address these challenges, we propose **Fragility-Aware Noise Training (FNT)** for robust intermediate-feature transmission in split computing over noisy channels. FNT identifies task-sensitive regions in intermediate features and applies stronger yet bounded perturbations to fragile locations, enabling targeted training. A lightweight ScoreNet guides adaptive noise injection, while joint optimization under varying bounded noise improves representation stability and fragility estimation. FNT is training-only, leaving the deployed architecture, feature size, and transmission path unchanged. The contributions of this work are as follows:

- We propose FNT, which dynamically identifies fragile regions in intermediate features and jointly optimizes ScoreNet within the Sensitivity Estimator and the split model in an adversarial manner, improving the robustness of intermediate-feature transmission in split computing.
- FNT demonstrates superior overall performance compared with prior robustness-enhancement methods, achieving an average accuracy gain of **8.75%** across all evaluated configurations. Under the strongest additive perturbations, the gain further increases to **36.27%**.
- FNT-trained models demonstrate strong cross-noise robustness beyond a single channel noise type, achieving an average accuracy gain of **10.75%** over the strongest baseline and a further improvement of **28.19%** under the strongest perturbations.

2 Related Work

We review prior work on deep learning robustness across input perturbations (Mintun et al., 2021), internal representations (Kim et al., 2021a), and communication processes (Shao et al., 2023). Existing robustness methods broadly follow three directions: random perturbation training (Trinh et al., 2024), adversarial and worst-case optimization (Sow et al., 2024; Losch et al., 2024), and communication-aware joint optimization (Shao et al., 2022). Despite differing in perturbation targets and optimization strategies, these methods share the goal of maintaining reliable predictions under non-ideal conditions.

SC partitions a DNN between resource-constrained edge devices and high-capacity cloud servers, enabling collaborative inference that balances computation, communication, and latency (Hao et al., 2026; Tan et al., 2025). Early work primarily focused on split-point selection to optimize deployment costs such as latency and energy consumption. However, partitioning exposes intermediate representations to real communication channels, where noise, transmission errors, and channel variations can degrade inference (Bajić et al., 2021). BottleNet++ was among the first to address this issue by combining feature compression with noise-aware training. More recent task-oriented communication methods further optimize transmitted representations under wireless channel effects to preserve task-relevant information for downstream inference (Shao et al., 2021; Wang et al., 2024).

Random perturbation training: COMSPLIT (COM) is a communication-aware split-learning method for heterogeneous IoT platforms (Ninkovic et al., 2024). It injects power-aware noise under predefined AWGN SNR levels during training, exposing intermediate features to communication perturbations without changing the split interface or feature dimensionality.

Adversarial training: Fast Gradient Method (FGM) is a robust training method that generates bounded adversarial perturbations from loss gradients (Goodfellow et al., 2014). It computes the task-loss gradient with respect to the perturbation target and generates a bounded perturbation along the normalized gradient direction for robust training.

Communication-aware joint optimization: Deep Joint Source-Channel Coding (DeepJSCC) is an end-to-end communication-aware framework for wireless transmission (Bourtsoulatze et al., 2019). It maps input representations to continuous complex-valued I/Q signals and jointly optimizes representation learning and transmission over noisy channels.

Existing robustness and communication-aware methods can partially alleviate performance degradation under non-ideal channels, but often overlook the non-uniform fragility of intermediate features across locations (Kim et al., 2021a). FNT learns task-loss-guided relative fragility scores and uses them to guide bounded local noise allocation. The estimator outputs fragility scores, which parameterize the perturbation distribution through score standardization, bounded SNR mapping, and stochastic sampling. Fragility estimation and robustness learning are jointly coupled through adversarial optimization.

3 Fragility-Aware Noise Training

This section introduces FNT, built on the bio-inspired "peripheral perception–neural transmission–central cognition" SC framework to improve the robustness of intermediate-feature transmission over noisy channels. As shown in Figure 2 and Algorithm 1, FNT follows three stages: fragility perception, noise allocation, and adversarial hardening. The Sensitivity Estimator, built around a lightweight ScoreNet, estimates relative feature fragility and uses the resulting scores to determine bounded local SNRs for fragility-aware noise injection. FNT jointly optimizes the split model and ScoreNet in a min–max manner, while the entire ScoreNet is used only during training.

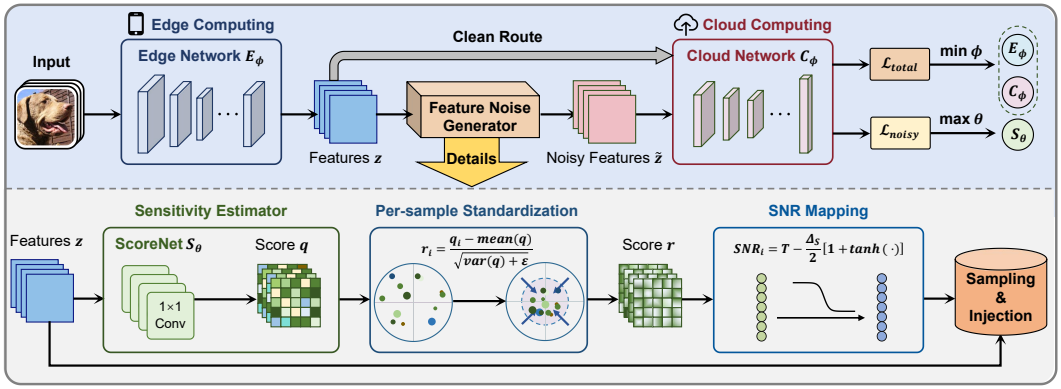


Figure 2: **Overview of FNT.** The top shows split-model training with FNT, while the bottom details its pipeline. Given intermediate features $\boldsymbol{z}$, the **Sensitivity Estimator** predicts sensitivity score $\boldsymbol{q}$, which is converted into relative fragility score $\boldsymbol{r}$ by **Per-sample Standardization**. The **SNR Mapping** module then maps $\boldsymbol{r}$ to local SNRs for adaptive noise sampling and injection, producing perturbed features $\tilde{\boldsymbol{z}}$ for robust training.

SC Model and Problem Formulation: Given $(x, y) \in \mathcal{D}$, the DNN is partitioned at a predefined split point into an edge network $E(\cdot;\phi_e)$ and a cloud network $C(\cdot;\phi_c)$, with $\boldsymbol{\phi} = \{\phi_e, \phi_c\}$ denoting the split-model parameters. The edge network extracts the intermediate feature for transmission:

$$z = E(x;\phi_e), \qquad z \in \mathbb{R}^{C\times H\times W} \tag{1}$$

where z denotes the transmitted intermediate feature. Under ideal transmission, the clean loss is

$$\mathcal{L}_{\text{clean}} = \text{CE}\big(C(z;\phi_c), y\big) \tag{2}$$

In practical communication, the transmitted intermediate feature may be corrupted by channel perturbations. The feature received by the cloud is modeled as

$$\widetilde{z} = z + \delta, \qquad \delta \sim \mathcal{P}_\xi \tag{3}$$

where δ denotes the channel perturbation and $\mathcal{P}_\xi$ is governed by an unknown channel state ξ. After receiving the perturbed intermediate feature, the cloud network computes the loss as:

$$\mathcal{L}_{\text{noisy}} = \text{CE}\big(C(\widetilde{z}; \phi_c), y\big) \tag{4}$$

Our goal is to achieve robust intermediate feature transmission under unknown and varying channel perturbations. Rather than exhaustively modeling diverse noise characteristics, FNT focuses on feature-level fragility and reinforces locations whose perturbations are more detrimental to task performance in this framework.

Relative Fragility Estimation: FNT first employs a **Sensitivity Estimator** to learn relative feature fragility. Within this module, a lightweight ScoreNet S_θ maps the intermediate feature z to a location-wise sensitivity score map:

$$q = S_\theta(z), \qquad q \in \mathbb{R}^{C\times H\times W}. \tag{5}$$

Here, q provides task-loss-guided proxies for relative feature fragility, reflecting the susceptibility of task performance to feature perturbations. A lightweight ScoreNet limits the capacity and computational overhead of the fragility estimator; its capacity is examined in Appendix D.3. To remove sample-wise score offsets and scale differences, we standardize q independently for each sample:

$$r_i = \frac{q_i - \text{mean}(q)}{\sqrt{\text{var}(q) + \epsilon_s}} \tag{6}$$

where i indexes the feature location and ϵ_s ensures numerical stability. The resulting r contains standardized relative fragility scores and serves as the basis for subsequent local SNR allocation. Appendix E.3 analyzes the learned fragility-related allocation patterns and examines the association between the fragility ranking and perturbation sensitivity.

Bounded Local SNR Allocation: FNT maps fragility scores r to bounded local SNRs, with lower SNRs inducing stronger training perturbations at more fragile locations. A reference SNR is sampled per mini-batch to model varying communication conditions:

$$T \sim \mathcal{U}(T_{\min}, T_{\max}) \tag{7}$$

We then map each fragility score r_i to a bounded allocation coefficient:

$$a_i = \frac{1}{2}\left[1 + \tanh\left(2(r_i - \kappa)\right)\right], \qquad a_i \in (0, 1) \tag{8}$$

where κ controls the mapping shift. A larger r_i produces a larger a_i, indicating that the corresponding location is more fragile. To ensure sufficiently strong perturbations for effective robustness training, we define the local SNR as:

$$\text{SNR}_i = T - \Delta_S a_i. \tag{9}$$

where Δ_S controls the local SNR span and thus the range of perturbation strengths. Within this bounded range, the location-wise noise allocation is adaptively determined by the estimated fragility, with more fragile features receiving lower SNRs and stronger perturbations.

Heteroscedastic Noise Injection and Adversarial Optimization: FNT converts local SNRs into location-dependent noise variances. Since feature scales vary across samples, we compute each sample's average feature power to determine these variances:

$$P_z = \frac{1}{CHW}\|z\|_F^2, \qquad \sigma_i^2 = P_z 10^{-\text{SNR}_i/10} \tag{10}$$

We assign the noise variance at each location according to its corresponding local SNR, thereby producing heteroscedastic perturbations.

$$\delta_i \sim \mathcal{N}(0, \sigma_i^2), \qquad \widetilde{z} = z + \delta \tag{11}$$

FNT performs adversarial joint optimization between the split model and the Sensitivity Estimator. The split model minimizes the joint loss over clean and perturbed features, defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{clean}} + \mathcal{L}_{\text{noisy}} \tag{12}$$

Meanwhile, the Sensitivity Estimator maximizes $\mathcal{L}_{\text{noisy}}$ to identify more challenging noise allocations. The overall optimization is formulated as:

$$\min_{\phi} \mathbb{E}[\mathcal{L}_{\text{total}}], \qquad \max_{\theta} \mathbb{E}[\mathcal{L}_{\text{noisy}}] \tag{13}$$

Through this min–max optimization, the Sensitivity Estimator progressively strengthens perturbations at fragile locations, while the split model learns more robust intermediate representations.

In summary, FNT integrates relative fragility estimation, bounded local SNR allocation, and adversarial optimization into a unified training framework. By using the learned fragility scores to guide heterogeneous noise injection, it encourages robust intermediate representations under varying channel perturbations.

4 EVALUATION

This section presents a comprehensive evaluation of FNT to validate its effectiveness in enhancing model robustness under the bio-inspired SC framework. Our evaluation focuses on two aspects: (1) comparisons with existing optimization methods across different tasks and model architectures; and (2) ablation studies on key FNT components. Method-specific perturbation settings and FNT hyperparameter analyses are provided in Appendix A.2, which further discusses the fairness of perturbation strengths: stronger noise does not inherently improve robustness, and the key is whether a method can stably accommodate stronger perturbations during training.

Table 1: **Robustness across tasks, architectures, and split points under AWGN perturbations.** We compare FNT with Vanilla training and prior robustness methods (COM, FGM, and DeepJSCC). Each entry reports the average accuracy across five SNR levels, with the worst-case accuracy shown in parentheses. Bold text indicates FNT results.

Dataset	**Model / sp**	**AWGN accuracy (%): Avg. ↑ (Worst ↑)**				
		Vanilla	**COM**	**FGM**	**DeepJSCC**	**FNT**
(a) Cross-task						
ImageNet-1K	ResNet-50 / sp3	58.60 (4.96)	73.12 (59.00)	59.86 (6.99)	61.43 (20.54)	**74.26 (68.76)**
Chest CT	ResNet-18 / sp2	43.43 (17.14)	44.53 (17.14)	50.11 (20.51)	55.53 (20.98)	**69.11 (66.44)**
UPMC Food-101	ViLT-B/32 / sp4	51.44 (1.92)	73.77 (7.60)	51.54 (1.90)	55.61 (1.79)	**88.00 (82.97)**
(b) Cross-architecture						
CIFAR-100	ResNet-18 / sp2	68.03 (39.45)	71.90 (55.51)	71.08 (51.06)	58.77 (27.79)	**74.55 (72.44)**
CIFAR-100	ResNet-50 / sp2	72.52 (56.28)	75.36 (68.25)	75.68 (69.13)	65.41 (45.09)	**76.03 (75.11)**
CIFAR-100	VGG-16 / sp23	58.25 (30.45)	63.16 (44.91)	60.24 (34.96)	55.82 (27.63)	**66.95 (61.71)**
(c) Cross-split						
CIFAR-10	MobileNetV2 / sp1	62.74 (18.82)	69.95 (16.71)	62.18 (12.33)	53.64 (12.22)	**85.92 (76.75)**
CIFAR-10	MobileNetV2 / sp3	58.07 (15.68)	68.77 (20.96)	69.35 (22.02)	56.91 (17.72)	**85.16 (78.53)**
CIFAR-10	MobileNetV2 / sp6	70.16 (29.68)	77.53 (44.11)	71.72 (28.17)	67.46 (32.94)	**88.73 (85.81)**

4.1 EXPERIMENTAL SETUP

Split point: We define **split point (sp)** as the boundary where features are transmitted from the edge to the cloud. We consider only complete module boundaries, such as residual blocks, and avoid cuts within individual operations or internal branches to preserve network integrity (Li et al., 2019). We use existing split-point selection strategies, with detailed partition configurations provided in Appendix B.2.

Datasets and models: We evaluate FNT across multiple tasks under diverse configurations, including CIFAR-10 (MobileNetV2, sp1/3/6), CIFAR-100 (ResNet18, sp2; ResNet50, sp2; VGG16-BN,

sp23), ImageNet-1K (ResNet50, sp3), Chest CT (ResNet18, sp2), and UPMC Food-101 (ViLT-B/32, sp4). Detailed dataset, architecture, and split configurations are provided in Appendix B.

Noise Settings and Evaluation Protocol: We denote the training and test perturbations as δ_{tr} and δ_{te}, respectively. Each method is trained using the perturbation distribution specified in its original work, with detailed implementations and parameter settings provided in Appendix A. We consider three representative test perturbation distributions $P_{\xi_{\mathrm{te}}}$: AWGN, packet erasure, and colored Gaussian noise (Shannon, 1949; Dana et al., 2006; Cover & Pombra, 1989). Each model is evaluated 10 times under each noise setting for reliable performance assessment. **Vanilla** denotes standard training without noise or robustness enhancement. All methods use identical base hyperparameters under the same experimental configuration.

4.2 Robustness Evaluation

In this section, we evaluate the robust generalization of FNT across tasks, architectures, split points, and channel conditions, which induce different intermediate representations and fragility patterns. We further conduct cross-noise evaluations to assess whether the learned robustness extends beyond the training noise distribution while keeping the data and base training settings unchanged. Table 1 summarizes the overall performance. Appendix E provides the complete numerical results and additional experiments. Appendix E.3 further analyzes the learned fragility-related allocation patterns and evaluates the fragility ranking through controlled perturbation interventions.

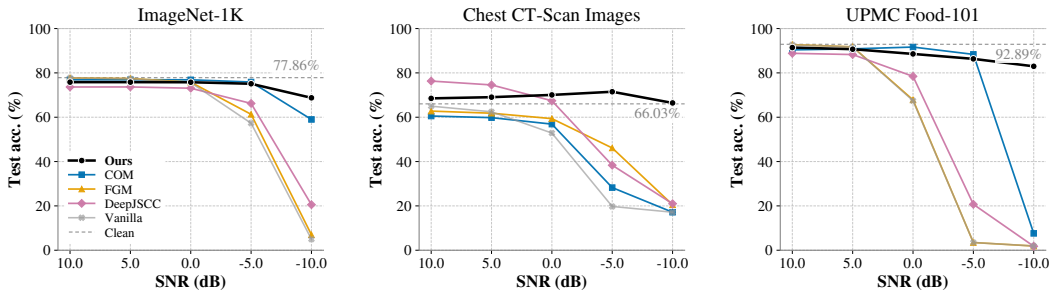


Figure 3: **Robustness comparison across different tasks.** FNT shows greater stability than competing methods as channel noise increases. Its robustness advantage becomes more pronounced under severe perturbations, maintaining around **70%** average accuracy even at **-10 dB**.

Robustness across Tasks: We evaluate FNT on natural image classification, medical imaging, and vision-language classification. The observed gains across these tasks support the applicability of fragility-aware training to the evaluated intermediate representations and data distributions. ImageNet-1K and UPMC Food-101 are fine-tuned from pretrained weights, while the Chest CT model is trained from scratch.

Under AWGN perturbation, FNT exhibits substantially slower performance degradation as channel noise increases (Fig. 3), with gray dashed lines indicating the clean accuracies. On ImageNet-1K, FNT retains **68.76%** accuracy under the strongest perturbation, exceeding COM (59.00%) by **9.76%** and Vanilla (4.96%) by **63.80%**, while remaining only 9.10% below the clean accuracy of 77.86%. On Chest CT, FNT achieves **66.44%**, compared with 20.98% for the strongest baseline and 17.14% for Vanilla. It also surpasses the clean reference of 66.03% at several noise levels, suggesting a regularization effect from bounded fragility-aware perturbations in this relatively small-data regime. On the multimodal UPMC Food-101 task, FNT maintains **82.97%** accuracy under the strongest perturbation, outperforming the best competing method (7.60%) by **75.37%**, further demonstrating its generalization to complex multimodal representations. Importantly, under mild noise, FNT stays within 2.5% of the clean reference across all three tasks.

Across the evaluated tasks, FNT uses learned relative fragility scores to guide local noise allocation. The observed robustness gains support the applicability of this fragility-aware training scheme to different tasks and modalities.

Robustness across Model Architectures: We fix the task to CIFAR-100 and vary only the backbone to examine whether FNT depends on a specific network architecture. Different architectures

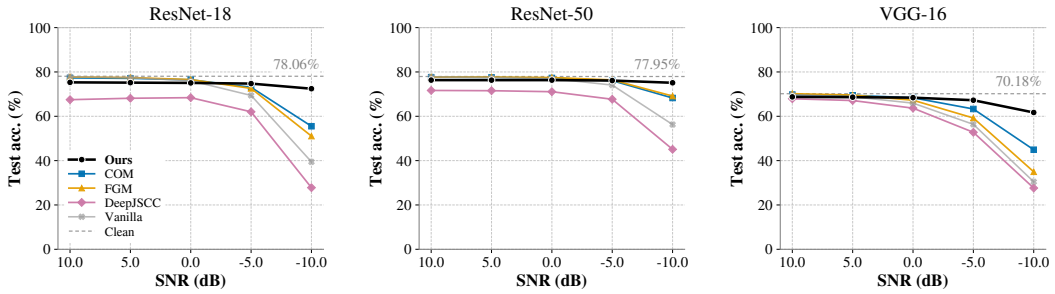


Figure 4: **Robustness across model architectures on CIFAR-100.** FNT remains robust across ResNet-18, ResNet-50, and VGG-16, with the accuracy degradation kept within **10%**, and consistently outperforming competing methods under strong channel noise.

yield distinct intermediate representations and sensitivity patterns, allowing us to isolate FNT's adaptability to architectural variations.

As shown in Fig. 4 and Table 1, FNT maintains strong robustness across all three architectures. On ResNet-18, FNT achieves an average accuracy of **74.55%** and retains **72.44%** at -10 dB, exceeding the strongest baseline (55.51%) by **16.93%** and Vanilla (39.45%) by **32.99%**. On ResNet-50, FNT reaches **75.11%** under the strongest perturbation, outperforming the best baseline (69.13%) by **5.98%**, while achieving the highest average accuracy of **76.03%**. A similar trend is observed on VGG-16, where FNT attains **61.71%** at -10 dB, compared with 44.91% for the strongest baseline and 30.45% for Vanilla, and improves the average accuracy to **66.95%**. Even under the strongest perturbation, FNT keeps the accuracy drop within **8.5%** of the clean reference across all three architectures. A similar pattern is observed under mild noise, where FNT remains within **3%** of the clean accuracy across all architectures despite its substantial gains under severe perturbations.

Overall, FNT uses the same relative-fragility estimation and SNR-mapping procedure across the evaluated architectures. The observed gains support the applicability of fragility-aware training to different network structures.

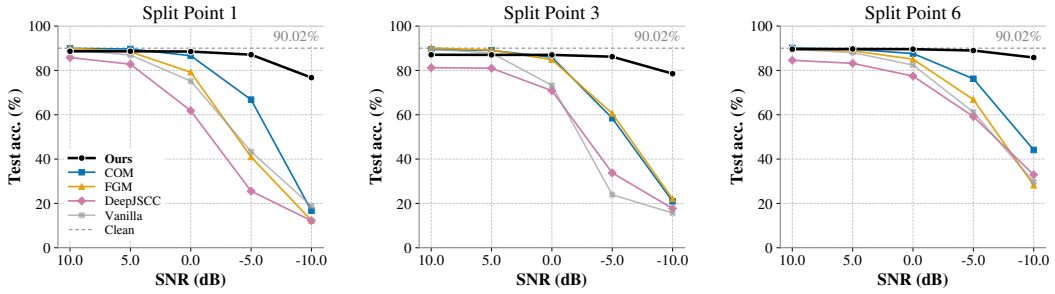


Figure 5: **Robustness across split points on CIFAR-10 with MobileNetV2.** With the task and backbone fixed, FNT remains robust across different sp, retaining at least **76.75%** accuracy and substantially outperforming competing methods.

Robustness across Split Points: Since the same partitioning strategy may yield different sp under different deployment settings, robustness across sp is important for practical SC. We fix the task and backbone to CIFAR-10 and MobileNetV2 and vary only the sp, which exposes representations at different depths with distinct feature and fragility patterns. Consistent performance across sp demonstrates FNT's adaptability to different edge–cloud partitions.

As shown in Fig. 5 and Table 1, FNT maintains strong robustness across all three split points, with average accuracies of **85.92%**, **85.16%**, and **88.73%** at sp1, sp3, and sp6, respectively. Under mild and moderate noise, FNT remains close to the clean reference and consistently matches or outperforms the competing methods, while the performance gap becomes increasingly evident as the channel condition deteriorates. Under the strongest perturbation, FNT retains **76.75%**, **78.53%**,

and **85.81%** accuracy at the three split points, whereas the strongest competing methods achieve only 18.82%, 22.02%, and 44.11%, yielding gains of **57.93%**, **56.51%**, and **41.70%**, respectively. These results show that FNT remains competitive under mild noise, while its advantage becomes increasingly pronounced as the channel condition deteriorates.

Overall,changing the split point alters the intermediate representation exposed to channel perturbations. In each evaluated configuration, FNT learns relative fragility scores at the selected split interface and uses them to guide local noise allocation. The results support the effectiveness of this fragility-aware training procedure across the tested edge–cloud partitions.

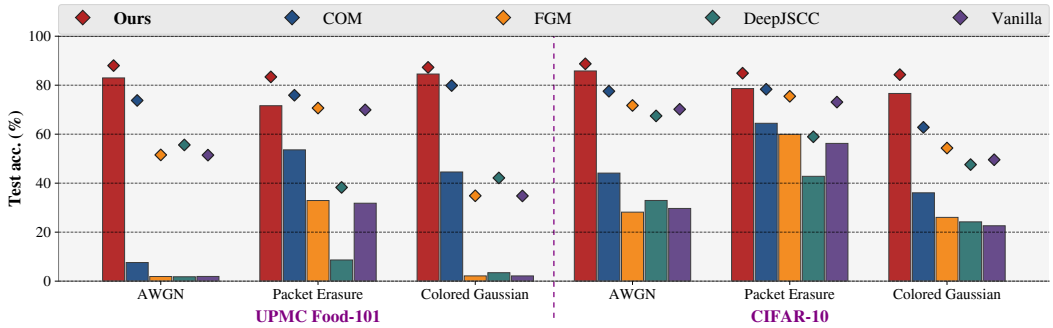


Figure 6: **Robustness across noises:** Diamonds denote the mean accuracy across all noise levels, while bars indicate the accuracy under the strongest perturbation. FNT consistently maintains around **80%** accuracy under severe noise and substantially outperforms competing methods across different channel conditions.

Robustness across Channel Conditions: Matched train–test noise settings provide only a limited view of deployment robustness, as practical communication systems are subject to diverse impairments, with recent work categorizing **15** representative transceiver non-idealities (Aggarwal & Bohara, 2026). We therefore evaluate FNT under three structurally distinct perturbations—AWGN, packet erasure, and colored Gaussian noise—which show low pairwise correlation and substantially different degradation patterns. Detailed perturbation definitions and distributional analyses are provided in Appendix C. After training, each model is evaluated under all three perturbation types to assess robustness generalization across heterogeneous channel conditions.

As shown in Fig. 6, FNT maintains strong robustness across all three channel perturbations despite their substantially different structures. Under AWGN, where independent Gaussian perturbations are applied element-wise, the results are consistent with the main benchmark: FNT achieves mean/worst-case accuracies of **88.00%/82.97%** on UPMC Food-101 and **88.73%/85.81%** on CIFAR-10, substantially outperforming the competing methods. When the perturbation changes to packet erasure, which removes entire feature packets rather than introducing additive noise, FNT still retains around **70–80%** worst-case accuracy on both tasks, although the performance gap becomes smaller than under AWGN. Under colored Gaussian noise, where perturbations are spatially correlated, FNT again maintains above **75%** worst-case accuracy and exhibits a pronounced advantage over the baselines. These results show that the robustness learned by FNT is not restricted to independent additive perturbations, but transfers to both structured feature loss and correlated noise.

Overall, FNT consistently maintains strong robustness across these perturbations, despite their substantially different degradation patterns. Its stable mean and worst-case performance across both tasks indicates that the learned robustness is not tied to a specific channel model, but generalizes effectively to heterogeneous communication conditions.

4.3 Ablation Studies

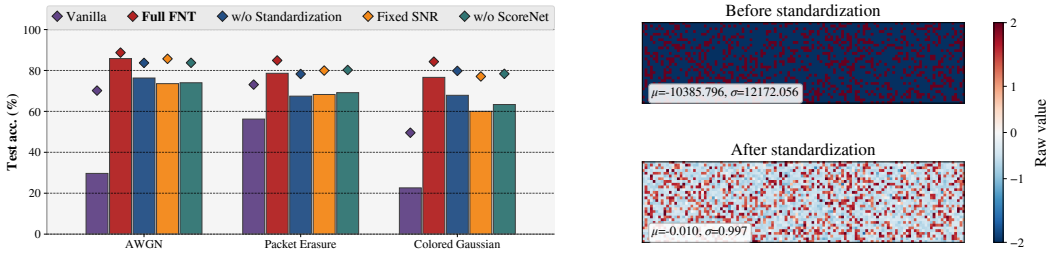
To verify that FNT's effectiveness stems from the joint contribution of its components rather than a single dominant module, we ablate ScoreNet, per-sample standardization, and reference SNR sampling, which control fragility estimation, score calibration, and noise-range coverage, respectively. Experiments are conducted on CIFAR-10 (MobileNetV2, sp6), as shown in Fig. 7 with additional analyses provided in Appendix D.

w/o Standardization: Removing per-sample standardization reduces the mean accuracy by over **4.5%** across all three perturbations, with the largest drop of **6.63%** under packet erasure. Under the strongest perturbations, the reduction exceeds **8.7%** and reaches **11.17%**. As shown in Fig. 7 (b), randomly sampled feature locations exhibit a highly polarized score distribution before standardization, causing the nonlinear SNR mapping to saturate near its two bounds. Standardization concentrates the scores into the transition region, enabling smoother and more differentiated SNR allocation. This prevents the allocation from collapsing to nearly binary perturbation strengths and improves the effectiveness of fragility-aware training.

Reference SNR: Fixing the reference SNR reduces the mean accuracy by at least **3.0%**, with the largest drop of **7.23%** under colored Gaussian noise. Under the strongest perturbations, the reduction exceeds **10%** for all three noise types and reaches **16.59%** for colored Gaussian noise. This larger degradation is likely due to its spatially correlated structure, where strong perturbations jointly affect neighboring features. Reference SNR sampling instead broadens the training-noise range, improving robustness to such severe correlated disturbances.

w/o ScoreNet: Removing the learnable ScoreNet reduces mean accuracy by over **4.6%**, with a maximum drop of **5.94%** under colored Gaussian noise. Under the strongest perturbations, the degradation exceeds **9.4%** and reaches **13.27%**. This performance gap supports the use of learned fragility scores over the feature-magnitude-based alternative for noise allocation in the evaluated setting.

Overall, all ablated variants still outperform Vanilla by at least **5%** in mean accuracy across the three perturbations, but consistently fall behind Full FNT, especially under severe noise. This confirms that FNT benefits from the joint effects of Sensitivity Estimator, score standardization, and diverse SNR sampling rather than any single component.



(a) Ablation study of the main FNT components.

(b) Standardization of Sensitivity Scores

Figure 7: (a) **Ablation of FNT components:** Removing standardization, random SNR sampling, or ScoreNet consistently reduces robustness, especially under severe noise. (b) **Effect of per-sample standardization.** Standardization produces a more concentrated sensitivity-score distribution.

5 Conclusion

Motivated by the hierarchical progression from peripheral perception through neural transmission to central cognition in biological systems, we present FNT to improve the robustness of intermediate-feature transmission over noisy channels. FNT learns relative feature fragility, maps the resulting scores to bounded local SNRs, and performs adversarial optimization under varying perturbation strengths, encouraging robust intermediate representations without modifying the deployed inference architecture. Extensive experiments across diverse tasks, architectures, split points, and channel conditions demonstrate that FNT significantly improves robustness, particularly under severe and unmatched perturbations. A limitation of FNT is that it does not directly optimize feature-transmission efficiency or modify the transmission process, and thus does not reduce communication cost or recover lost information. This limitation reflects its training-only design, which preserves the deployed split architecture without introducing additional online communication modules.

AI USE STATEMENT

Generative AI tools were used primarily to assist with code debugging and language editing. All research methodology, experimental results, analyses, and scientific conclusions were developed, reviewed, and verified by the authors. The authors take full responsibility for the content of this work.

REPRODUCIBILITY STATEMENT

All figures, tables, and numerical results reported in this paper are obtained from actual experimental runs. For reproducibility, the Appendix provides the complete experimental configurations, including dataset and model settings, edge–cloud split points, training hyperparameters, baseline implementations, channel perturbation models, and the parameter settings of FNT. Following these configurations allows the reported experiments to be reproduced and the presented results to be recovered within normal stochastic variations across training and channel realizations. Upon acceptance, we will publicly release the source code and implementation details to further facilitate reproducibility.

REFERENCES

Parag Aggarwal and Vivek Ashok Bohara. From legacy to future 6g network: Challenges and mitigation techniques of wireless transceivers non-idealities. *IEEE Open Journal of the Communications Society*, 2026.

Ivan V Bajić, Weisi Lin, and Yonghong Tian. Collaborative intelligence: Challenges and opportunities. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8493–8497. IEEE, 2021.

Eirina Bourtsoulatze, David Burth Kurka, and Deniz Gündüz. Deep joint source-channel coding for wireless image transmission. *IEEE Transactions on Cognitive Communications and Networking*, 5(3):567–579, 2019.

Zibo Chen, Ruxin Li, and Zilu Wang. Diversity-aware weight perturbation promotes robust adaptation. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=5VoK9nUVSD.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pp. 1310–1320. PMLR, 2019.

Thomas M Cover and Sandeep Pombra. Gaussian feedback capacity. *IEEE Transactions on Information Theory*, 35(1):37–43, 1989.

Amir F Dana, Radhika Gowaikar, Ravi Palanki, Babak Hassibi, and Michelle Effros. Capacity of wireless erasure networks. *IEEE Transactions on Information Theory*, 52(3):789–804, 2006.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Ranjie Duan, YueFeng Chen, Yao Zhu, Xiaojun Jia, Rong Zhang, et al. Inequality phenomenon in ℓ_∞-adversarial training, and its unrealized threats. In *The Eleventh International Conference on Learning Representations*, 2023.

Amir Erfan Eshratifar, Mohammad Saeed Abrishami, and Massoud Pedram. Jointdnn: An efficient training and inference engine for intelligent mobile cloud computing services. *IEEE transactions on mobile computing*, 20(2):565–576, 2019.

A Aldo Faisal, Luc PJ Selen, and Daniel M Wolpert. Noise in the nervous system. *Nature reviews neuroscience*, 9(4):292–303, 2008.

Angélica Pérez Fornos, Raymond Van de Berg, Jörg Sommerhalder, and Nils Guinand. Designing artificial senses: steps from physiology to clinical implementation. *Swiss Medical Weekly*, 149 (2122):w20061–w20061, 2019.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Yuntao Hao, Nan Ding, Weiguo Xia, Hongwei Ge, and Li Xu. Dnn partitioning for cooperative inference in edge intelligence: Modeling, solutions, toolchains. *ACM Computing Surveys*, 58(8): 1–34, 2026.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

Chuang Hu, Wei Bao, Dan Wang, and Fengming Liu. Dynamic adaptive dnn surgery for inference acceleration on the edge. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pp. 1423–1431. IEEE, 2019.

Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor Mudge, Jason Mars, and Lingjia Tang. Neurosurgeon: Collaborative intelligence between the cloud and mobile edge. *ACM SIGARCH Computer Architecture News*, 45(1):615–629, 2017.

Junho Kim, Byung-Kwan Lee, and Yong Man Ro. Distilling robust and non-robust features in adversarial examples by information bottleneck. *Advances in Neural Information Processing Systems*, 34:17148–17159, 2021a.

Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *International conference on machine learning*, pp. 5583–5594. PMLR, 2021b.

Jong Hwan Ko, Taesik Na, Mohammad Faisal Amir, and Saibal Mukhopadhyay. Edge-host partitioning of deep neural networks with feature space encoding for resource-constrained internet-of-things platforms. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–6. IEEE, 2018.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Stefanos Laskaridis, Stylianos I Venieris, Mario Almeida, Ilias Leontiadis, and Nicholas D Lane. Spinn: Synergistic progressive inference of neural networks over device and cloud. In *Proceedings of the 26th annual international conference on mobile computing and networking*, pp. 1–15, 2020.

En Li, Liekang Zeng, Zhi Zhou, and Xu Chen. Edge ai: On-demand accelerating deep neural network inference via edge computing. *IEEE transactions on wireless communications*, 19(1): 447–457, 2019.

Xian Li, Suzhi Bi, Shuoyao Wang, Xiangcheng Li, and Ying-Jun Angela Zhang. Digital semantic device-edge co-inference with task-oriented arq. *IEEE Transactions on Vehicular Technology*, 73 (9):13986–13990, 2024.

Ji Lin, Wei-Ming Chen, Yujun Lin, Chuang Gan, Song Han, et al. Mcunet: Tiny deep learning on iot devices. *Advances in neural information processing systems*, 33:11711–11722, 2020.

Hao Liu, Mohammed E Fouda, Ahmed M Eltawil, and Suhaib A Fahmy. Split dnn inference for exploiting near-edge accelerators. In *2024 IEEE International Conference on Edge Computing and Communications (EDGE)*, pp. 84–91. IEEE, 2024.

Max Losch, Mohamed Omran, David Stutz, Mario Fritz, and Bernt Schiele. On adversarial training without perturbing all examples. In *International Conference on Learning Representations*, volume 2024, pp. 47995–48015, 2024.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Yoshitomo Matsubara, Marco Levorato, and Francesco Restuccia. Split computing and early exiting for deep learning applications: Survey and research challenges. *ACM Computing Surveys*, 55(5): 1–30, 2022.

Eric Mintun, Alexander Kirillov, and Saining Xie. On interaction between augmentations and corruptions in natural corruption robustness. *Advances in Neural Information Processing Systems*, 34:3571–3583, 2021.

Vukan Ninkovic, Dejan Vukobratovic, Dragisa Miskovic, and Marco Zennaro. Comsplit: A communication-aware split learning design for heterogeneous iot platforms. *IEEE Internet of Things Journal*, 12(5):5305–5319, 2024.

Claire Ribrault, Ken Sekimoto, and Antoine Triller. From the stochasticity of molecular processes to the variability of synaptic transmission. *Nature Reviews Neuroscience*, 12(7):375–387, 2011.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 4510–4520. Ieee, 2018.

Claude E Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, 1949.

Jiawei Shao and Jun Zhang. Bottlenet++: An end-to-end approach for feature compression in device-edge co-inference systems. In *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6. IEEE, 2020.

Jiawei Shao, Yuyi Mao, and Jun Zhang. Learning task-oriented communication for edge inference: An information bottleneck approach. *IEEE Journal on Selected Areas in Communications*, 40(1): 197–211, 2021.

Jiawei Shao, Yuyi Mao, and Jun Zhang. Task-oriented communication for multidevice cooperative edge inference. *IEEE Transactions on Wireless Communications*, 22(1):73–87, 2022.

Jiawei Shao, Xinjie Zhang, and Jun Zhang. Task-oriented communication for edge video analytics. *IEEE Transactions on Wireless Communications*, 23(5):4141–4154, 2023.

Mohammad Q. Shatnawi, Qusai Abuein, and Romesaa Al-Quraan. Deep learning-based approach to diagnose lung cancer using ct-scan images. *Intelligence-Based Medicine*, 11:100188, 2025. ISSN 2666-5212. doi: https://doi.org/10.1016/j.ibmed.2024.100188. URL `https://www.sciencedirect.com/science/article/pii/S2666521224000553`.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Daouda Sow, Sen Lin, Zhangyang Wang, and Yingbin Liang. Doubly robust instance-reweighted adversarial training. In *International Conference on Learning Representations*, volume 2024, pp. 1274–1300, 2024.

Jingwei Tan, Fagui Liu, Bin Wang, Qingbo Wu, and CL Philip Chen. Ec5: Edge–cloud collaborative computing framework with compressive communication. *Future Generation Computer Systems*, 166:107715, 2025.

Trung Trinh, Markus Heinonen, Luigi Acerbi, and Samuel Kaski. Improving robustness to corruptions with multiplicative weight perturbations. *Advances in Neural Information Processing Systems*, 37:35492–35516, 2024.

Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

Mengyang Wang, Jiahui Li, Mengyao Ma, and Xiaopeng Fan. Snn-sc: A spiking semantic communication framework for collaborative intelligence. *IEEE Transactions on Vehicular Technology*, 74(4):5883–5896, 2024.

Xin Wang, Devinder Kumar, Nicolas Thome, Matthieu Cord, and Frederic Precioso. Recipe recognition with large multimodal food dataset. In *2015 IEEE International conference on multimedia & expo workshops (ICMEW)*, pp. 1–6. IEEE, 2015.

Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 501–509, 2019.

A Experimental Configurations

A.1 Baseline Methods

A.1.1 COM

Following the communication-aware training strategy of COMSPLIT (Ninkovic et al., 2024), COM introduces AWGN with different perturbation strengths during intermediate-feature transmission, enabling the split model to adapt to heterogeneous channel conditions. Specifically, each feature location is assigned one of two predefined SNR states, where a higher SNR corresponds to weaker noise and a lower SNR to stronger noise, with their relative occurrence controlled by a mixing ratio. In this work, we retain this two-level noise training mechanism while excluding the early-exit, heterogeneous-device, and communication-resource management components of the original COMSPLIT framework.

Mathematical Modeling: Given the transmitted intermediate feature $z \in \mathbb{R}^{C\times H\times W}$, we first compute its average feature power as

$$P_z = \frac{1}{CHW}\|z\|_F^2. \tag{A.1}$$

For each feature location i, COM assigns one of two predefined SNR states:

$$s_i = \begin{cases} s_{\text{weak}}, & m_i = 0, \\ s_{\text{strong}}, & m_i = 1, \end{cases} \qquad m_i \sim \text{Bernoulli}(\rho), \tag{A.2}$$

where $s_{\text{strong}} = s_{\text{weak}} - 5$ dB denotes the stronger noise condition, and ρ controls the proportion of strongly perturbed locations.

The noise variance at each location is determined by its assigned SNR:

$$\sigma_i^2 = P_z\, 10^{-s_i/10}, \tag{A.3}$$

and the perturbed feature is obtained by

$$n_i \sim \mathcal{N}(0, \sigma_i^2), \qquad \widetilde{z}_i = z_i + n_i. \tag{A.4}$$

The split model is then optimized using the noisy task loss:

$$\min_{\theta_e,\theta_c} \mathbb{E}_{(x,y),\,m,\,n}\left[\ell_{\text{CE}}\big(f_{\theta_c}(\widetilde{z}), y\big)\right], \tag{A.5}$$

where $z = f_{\theta_e}(x)$.

Key Parameters: The main parameters of COM are the weak-noise SNR s_{weak}, the strong-noise SNR s_{strong}, and the strong-noise ratio ρ. Following the settings considered in the original COM-SPLIT framework, we evaluate multiple COM configurations with different two-level SNR settings. For example, $s_{\text{strong}} \in \{0, -5, -10\}$ dB, with $s_{\text{weak}} = s_{\text{strong}} + 5$ dB. Since different configurations favor different channel conditions, we do not use a single fixed COM setting for all experiments. Instead, for each comparison, we select the configuration that achieves the best overall performance on the validation set.

A.1.2 FGM

Following the gradient-based adversarial training strategy (Goodfellow et al., 2014), FGM introduces task-aware perturbations into the transmitted intermediate features. Unlike random channel noise, the perturbation direction is determined by the gradient of the task loss with respect to the current split feature, thereby targeting locally sensitive feature directions. In this work, we adapt the fast gradient principle to the intermediate-feature space and employ a single-step L_2-normalized perturbation, rather than the input-space L_∞ sign perturbation used in the original formulation.

Mathematical Modeling: Given the transmitted intermediate feature $z = f_{\theta_e}(x)$, we first compute the gradient of the task loss with respect to z:

$$g_z = \nabla_z \ell_{\text{CE}}\big(f_{\theta_c}(z), y\big). \tag{A.6}$$

The feature-space adversarial perturbation is then constructed by normalizing the gradient under the L_2 norm:

$$\delta_{\text{FGM}} = \epsilon \frac{g_z}{\|g_z\|_2 + \varepsilon_g}, \tag{A.7}$$

where ϵ controls the perturbation magnitude and ε_g is a small constant for numerical stability. The perturbed intermediate feature is given by

$$\widetilde{z} = z + \delta_{\text{FGM}}. \tag{A.8}$$

The split model is optimized using the task loss evaluated on the adversarially perturbed feature:

$$\min_{\theta_e, \theta_c} \mathbb{E}_{(x,y)} \left[\ell_{\text{CE}}\big(f_{\theta_c}(\widetilde{z}), y\big) \right]. \tag{A.9}$$

Key Parameters: The primary parameter of FGM is the feature-space perturbation radius ϵ. In all experiments, we set $\epsilon = 0.1$ and employ a single-step L_2-normalized perturbation directly on the split feature. No random initialization, multi-step optimization, or projection procedure is used. The model is trained using the adversarial cross-entropy loss. Note that ϵ defines the perturbation radius in feature space and does not correspond to a communication SNR. In our experiments, $\epsilon = 0.1$ is already close to the strongest perturbation that FGM can stably accommodate. Because FGM follows the task-loss gradient, increasing ϵ further amplifies perturbations along the most sensitive feature directions, causing excessive corruption of vulnerable feature locations and destabilizing optimization. At $\epsilon = 0.15$, training fails to converge, indicating that $\epsilon = 0.1$ is near the practical upper perturbation strength for FGM under our setting.

A.1.3 DEEPJSCC

DeepJSCC employs a learnable joint source–channel coding strategy to directly map source representations into continuous channel symbols, allowing the encoding, transmission, and decoding processes to be jointly optimized (Bourtsoulatze et al., 2019). In this work, we adapt DeepJSCC to intermediate-feature transmission in split computing. A learnable encoder is inserted after the edge-side network to compress and map the split feature into continuous I/Q channel representations. After transmission through a noisy channel, a corresponding decoder reconstructs the intermediate representation before it is processed by the cloud-side network.

Mathematical Modeling: Given the transmitted intermediate feature $z = f_{\theta_e}(x)$, the channel encoder E_ψ maps it into a continuous representation:

$$u = E_\psi(z) \in \mathbb{R}^{2K \times H \times W}, \tag{A.10}$$

where pairs of encoded channels represent the in-phase and quadrature components of K complex-valued channel symbols.

The average transmitted power is computed as

$$P_u = \frac{1}{KHW} \sum_{k,h,w} \left[\left(u^{(I)}_{k,h,w} \right)^2 + \left(u^{(Q)}_{k,h,w} \right)^2 \right], \tag{A.11}$$

and the encoded representation is power-normalized as

$$x_c = \frac{u}{\sqrt{P_u + \varepsilon_c}}, \tag{A.12}$$

where ε_c is a small constant for numerical stability.

The normalized channel representation is transmitted through a complex Gaussian channel:

$$y_c = x_c + n_c, \qquad n_c \sim \mathcal{CN}(0, \sigma_c^2 I). \tag{A.13}$$

The received representation is then reconstructed by the channel decoder D_ω:

$$\widetilde{z} = D_\omega(y_c). \tag{A.14}$$

The edge model, channel encoder, channel decoder, and cloud model are jointly optimized using

$$\min_{\theta_e, \psi, \omega, \theta_c} \mathbb{E}_{(x,y),\, n_c} \left[\ell_{\mathrm{CE}}\big(f_{\theta_c}(\widetilde{z}), y \big) \right]. \tag{A.15}$$

Key Parameters: The main parameters of DeepJSCC are the training SNR, channel ratio, and encoder–decoder architecture. We use a training SNR of 10 dB and a channel ratio of $1/6$ throughout the main comparisons. Since DeepJSCC compresses the split feature into a lower-dimensional channel representation before transmission, excessively strong channel noise can severely corrupt this compressed bottleneck and lead to rapid performance degradation. We therefore use a moderate training SNR to maintain a meaningful encoded representation while still exposing the model to channel perturbations. For CNN-based models, both the encoder and decoder are implemented using 1×1 convolutions, with Tanh activation used in the channel encoder. Given a split feature with C channels and channel ratio r, the number of complex channel symbols is determined by

$$K = \max\left(1, \left\lfloor \frac{Cr}{2} \right\rfloor \right), \tag{A.16}$$

resulting in $2K$ real-valued encoded channels. The transmitted representation is power-normalized with $\varepsilon_c = 10^{-8}$. Unlike COM and FGM, where perturbations are applied directly to the original split feature, the communication noise in DeepJSCC is applied to the encoded channel representation.

In our experiments, a training SNR of 10 dB is already close to the strongest perturbation that DeepJSCC can stably accommodate. DeepJSCC compresses the split feature into a compact channel representation, which reduces dimensional exposure to channel noise but also concentrates task-relevant information into fewer transmitted symbols. Consequently, further lowering the training SNR causes stronger corruption of these information-dense symbols and makes joint encoder–decoder optimization difficult to converge. Thus, 10 dB is near the practical upper perturbation strength for DeepJSCC under our setting.

A.1.4 Perturbation-Strength Fairness and Training Stability

FNT is trained under a stronger perturbation regime than the competing methods. Specifically, the reference SNR is sampled from $[-10, -3]$ dB, and with $\Delta_S = 4$ dB, the resulting local SNRs can span $[-14, -3]$ dB. However, stronger training noise does not inherently guarantee better robustness. FNT combines per-sample score standardization with bounded fragility-aware SNR mapping to control local training-noise strengths, assigning stronger perturbations to locations with higher estimated fragility.

To examine training behavior under stronger perturbations, we conduct an additional **strong-training-noise stress test**, as shown in Table A.1, by substantially increasing the perturbation

Table A.1: Robustness under stronger training perturbations on CIFAR-10 with MobileNetV2 split6.

Test SNR (dB)	Vanilla	COM	FGM	DeepJSCC	FNT
-10	29.68	59.11	9.74	10.21	**85.81**
-5	61.20	77.28	10.18	9.83	**88.97**
0	82.42	80.12	10.36	10.42	**89.58**
5	88.05	82.23	9.91	9.68	**89.66**
10	89.45	83.12	10.12	10.07	**89.60**
Avg.	**70.16**	**76.37**	**10.06**	**10.04**	**88.73**

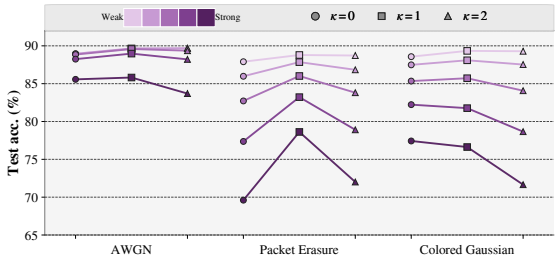
strength of each competing method while retaining its original perturbation mechanism. Specifically, for COM, we set the strong- and weak-noise SNRs to $s_{\text{strong}} = -14$ dB and $s_{\text{weak}} = -9$ dB, respectively; for FGM, we increase the feature-space perturbation radius from $\epsilon = 0.1$ to $\epsilon = 0.2$; and for DeepJSCC, we reduce the training channel SNR from 10 dB to 0 dB. The results show that stronger training noise does not provide consistent robustness gains. COM improves under the most severe test noise but degrades under moderate and mild conditions, while FGM and DeepJSCC become unstable and approach near-random performance. In contrast, FNT maintains high accuracy across the full test-SNR range. These results show that increasing perturbation strength alone does not consistently improve the tested baselines, while FNT maintains strong performance with its fragility-aware training scheme.

A.2 FNT Parameter Settings

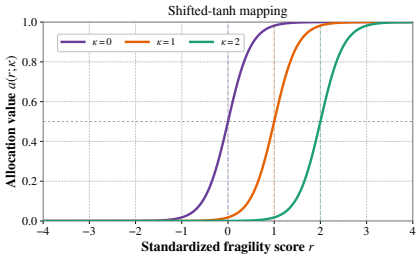
Algorithm 1 Fragility-Aware Noise Training

Input: training set $\mathcal{D}$; edge network $E(\cdot;\phi_e)$; cloud network $C(\cdot;\phi_c)$; Sensitivity Estimator S_θ; $T_{\min}, T_{\max}, \kappa, \Delta_S$; learning rates η_ϕ, η_θ
Output: trained split-model parameters $\phi = \{\phi_e, \phi_c\}$
begin
  Initialize ϕ_e, ϕ_c, and θ
  for $e = 1$ **to** N_{epoch} **do**
    foreach *mini-batch* $(x, y) \sim \mathcal{D}$ **do**
      $z \leftarrow E(x;\phi_e)$
      $q \leftarrow S_\theta(z)$
      $r \leftarrow \dfrac{q - \text{mean}(q)}{\sqrt{\text{var}(q) + \epsilon_s}}$
      $T \sim \mathcal{U}(T_{\min}, T_{\max})$
      $a \leftarrow \dfrac{1}{2}\left[1 + \tanh\left(2(r - \kappa)\right)\right]$
      $\text{SNR} \leftarrow T - \Delta_S a$
      $P_z \leftarrow \dfrac{1}{CHW}\|z\|_F^2$
      $\sigma^2 \leftarrow P_z 10^{-\text{SNR}/10}$
      $\epsilon \sim \mathcal{N}(0, I), \quad \delta \leftarrow \sqrt{\sigma^2} \odot \epsilon$
      $\widetilde{z} \leftarrow z + \delta$
      $\mathcal{L}_{\text{clean}} \leftarrow \text{CE}\big(C(z;\phi_c), y\big)$
      $\mathcal{L}_{\text{noisy}} \leftarrow \text{CE}\big(C(\widetilde{z};\phi_c), y\big)$
      $\mathcal{L}_{\text{total}} \leftarrow \mathcal{L}_{\text{clean}} + \mathcal{L}_{\text{noisy}}$
      $\phi \leftarrow \phi - \eta_\phi \nabla_\phi \mathcal{L}_{\text{total}}$
      $\theta \leftarrow \theta + \eta_\theta \nabla_\theta \mathcal{L}_{\text{noisy}}$
    end
  end
  Discard S_θ and **return** ϕ
end

A.2.1 SENSITIVITY TO THE MAPPING SHIFT



(a) Robustness under Different κ Values



(b) Effect of κ on SNR Allocation

Figure A.1: (a) **Robustness under different κ values.** Test accuracy of FNT with $\kappa \in \{0, 1, 2\}$ under AWGN, packet erasure, and colored Gaussian noise across different perturbation strengths. The results show that $\kappa = 1$ provides consistently strong performance across heterogeneous perturbation conditions. (b) **Effect of κ on SNR allocation.** The shifted-tanh mapping under different κ values. Increasing κ shifts the transition region toward larger fragility scores, making strong-noise allocation more selective to feature locations with high estimated fragility.

In FNT, κ controls the threshold of the shifted-tanh mapping used for local SNR allocation. Specifically, the standardized fragility score r_i is mapped to the allocation weight a_i as

$$a_i = \frac{1}{2}\left[1 + \tanh(2(r_i - \kappa))\right]. \tag{A.17}$$

As illustrated in Fig. A.1(b), varying κ horizontally shifts the mapping along the fragility-score axis and therefore directly controls the activation threshold and coverage of strong-noise allocation. When κ is too small, relatively low fragility scores can already receive large allocation weights, causing strong perturbations to cover a broad range of feature locations. In contrast, when κ is too large, strong perturbations become overly selective and may leave some locations with moderately high fragility scores insufficiently perturbed. An intermediate κ therefore provides a more balanced allocation by avoiding both overly broad and overly sparse strong-noise coverage. Based on this consideration, we evaluate $\kappa \in \{0, 1, 2\}$ to examine the sensitivity of FNT to the mapping shift and to identify a suitable setting.

As shown in Fig. A.1(a), we evaluate FNT with $\kappa \in \{0, 1, 2\}$ under AWGN, packet erasure, and colored Gaussian noise, where darker curves indicate stronger perturbation levels. Under relatively weak perturbations, different κ settings generally yield similar performance, while their differences become more pronounced as the perturbation strength increases. Overall, an excessively small κ tends to make strong-noise allocation too broad, whereas an excessively large κ makes the allocation overly selective, both of which can reduce robustness under severe perturbations. These results suggest that a moderate mapping threshold provides a better balance between perturbation coverage and selectivity. Based on this observation, we use $\kappa = 1$ as the default setting in FNT.

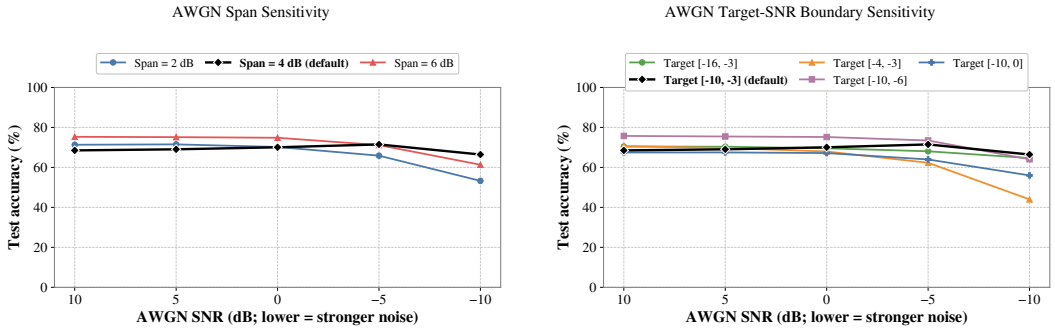
A.2.2 SENSITIVITY TO THE LOCAL SNR SPAN

In FNT, the local SNR span Δ_S controls the difference in perturbation strength across feature locations. Given the sampled reference SNR T, the local SNR at location i is determined by

$$\mathrm{SNR}_i = T - \Delta_S a_i, \tag{A.18}$$

where $a_i \in [0, 1]$ is determined by the estimated relative fragility of the corresponding feature location. A small Δ_S limits the variation in local SNRs, making the perturbation allocation closer to uniform noise training even when different locations have different fragility scores. Increasing Δ_S enlarges the local SNR contrast and assigns stronger perturbations to locations with higher estimated fragility. However, an excessively large span also increases the strongest local perturbation and may make robustness training overly aggressive. We therefore evaluate different values of Δ_S to examine the trade-off between fragility-aware perturbation differentiation and training robustness.

As shown in Fig. A.2a, we compare $\Delta_S \in \{2, 4, 6\}$ dB under different AWGN conditions. A small SNR span performs comparably under relatively mild perturbations but exhibits a more pronounced



(a) Sensitivity to the local SNR span Δ_S.

(b) Sensitivity to the reference SNR boundaries.

Figure A.2: **Sensitivity analysis of the SNR allocation parameters.** (a) Robustness of FNT under different local SNR spans Δ_S. (b) Robustness under different target-SNR sampling intervals $[S_{\min}, S_{\max}]$. The results illustrate the influence of the local perturbation range and the global training-noise range on robustness across AWGN conditions.

degradation as the test noise becomes stronger, indicating that insufficient local differentiation limits robustness under severe conditions. In contrast, a large span provides strong performance under mild and moderate perturbations but becomes less favorable as the channel condition further deteriorates. Overall, a moderate local SNR span provides a better balance across different perturbation strengths. Based on these results, we set $\Delta_S = 4$ dB as the default value in FNT.

A.2.3 Sensitivity to the Reference-SNR Boundaries

FNT samples a reference SNR T from a predefined interval:

$$T \sim \mathcal{U}(T_{\min}, T_{\max}), \tag{A.19}$$

which determines the range of global channel conditions encountered during training. The sampled reference SNR is further converted into location-dependent SNRs through the fragility-aware mapping

$$\mathrm{SNR}_i = T - \Delta_S a_i. \tag{A.20}$$

Since $a_i \in [0, 1]$, the local SNRs are bounded by

$$\mathrm{SNR}_i \in [T_{\min} - \Delta_S,\ T_{\max}]. \tag{A.21}$$

Therefore, $T_{\min}$ and $T_{\max}$ jointly determine the overall range of channel conditions covered during training. A higher $T_{\min}$ may provide insufficient exposure to severe perturbations, whereas an excessively low $T_{\min}$ places greater emphasis on highly corrupted conditions. Similarly, $T_{\max}$ determines the extent to which relatively mild channel conditions are included during training. We therefore vary the reference-SNR boundaries to examine how the training-noise range affects robustness across different test conditions.

As shown in Fig. A.2b, the choice of reference-SNR boundaries has a clear influence on robustness across different AWGN strengths. A relatively narrow or mild training range can maintain good performance under weak perturbations but degrades more noticeably as the test noise becomes stronger, indicating limited robustness outside the covered noise range. Expanding the interval toward stronger perturbations improves robustness under severe channel conditions, while excessively biasing the training range toward either mild or strong noise does not provide consistently favorable performance across all test strengths. Overall, an appropriately broad reference-SNR interval provides a better balance across heterogeneous channel conditions. Based on this observation, we use $[T_{\min}, T_{\max}] = [-10, -3]$ dB as the default setting in FNT.

B Datasets, Model Architectures, and Split-Point Selection

B.1 Datasets

CIFAR-10. CIFAR-10 is a widely used image classification benchmark consisting of $60{,}000$ RGB images of size 32×32 from 10 object categories, with $50{,}000$ images for training and $10{,}000$ for testing (Krizhevsky, 2009).

CIFAR-100. CIFAR-100 follows the same image size and dataset scale as CIFAR-10 but contains 100 fine-grained object categories, making it a more challenging classification benchmark (Krizhevsky, 2009).

ImageNet-1K. ImageNet-1K is the 1,000-class image classification subset used in the ILSVRC benchmark and contains large-scale natural images with substantial intra-class and inter-class diversity (Deng et al., 2009; Russakovsky et al., 2015).

Chest CT-Scan Images. The Chest CT-Scan Images dataset contains two-dimensional chest CT images from four categories: adenocarcinoma, large cell carcinoma, squamous cell carcinoma, and normal. We formulate it as a four-class CT image classification task (Shatnawi et al., 2025).

UPMC Food-101. UPMC Food-101 is a large-scale multimodal food recognition dataset containing 101 food categories with paired food images and associated textual information (Wang et al., 2015). Unlike the standard image-only Food-101 benchmark, UPMC Food-101 provides multimodal image–text samples collected from web sources.

B.2 Architectures

This work does not focus on edge–cloud partitioning strategies or optimal split-point search. Instead, we adopt the layer-wise partitioning paradigm commonly used in existing split computing studies, where the backbone is divided into an edge-side prefix and a cloud-side suffix, with intermediate features transmitted at the split point. To preserve architectural integrity and ensure controlled comparisons, all split points are placed at complete architectural boundaries rather than inside individual modules. Specifically, MobileNetV2 is split after complete inverted-residual blocks, ResNet after complete residual stages, VGG16-BN after pooling stages, and ViLT after complete Transformer blocks. The cited split-computing studies provide the general partitioning principle, while the exact split-point indices used in our experiments are fixed experimental settings rather than hardware-specific optima obtained through profiling or automatic partition search.

Table B.1: **Model architectures and edge–cloud partition settings.** The model references indicate the original backbone architectures, while the split references provide the layer-/module-wise partitioning principles. The exact split points are fixed experimental settings used in this work.

Item	Configuration
CIFAR-10 / MobileNetV2 / sp1	
Dataset	CIFAR-10
Model	MobileNetV2 (Sandler et al., 2018)
Edge-side	`features[0:2]`, including the initial convolution and the first complete inverted-residual block; **1,824 parameters**.
Cloud-side	`features[2:]` followed by global average pooling and the classifier; **2,234,858 parameters**.
Split point	sp1, located after a complete inverted-residual block.
Split reference	Layer-/feature-level partitioning following (Kang et al., 2017; Ko et al., 2018).
CIFAR-10 / MobileNetV2 / sp3	
Dataset	CIFAR-10
Model	MobileNetV2 (Sandler et al., 2018)

Continued on next page

Table B.1 – continued from previous page

Item	Configuration
Edge-side	`features[0:4]`, ending after a complete inverted-residual block; **15,792 parameters**.
Cloud-side	`features[4:]` followed by global average pooling and the classifier; **2,220,890 parameters**.
Split point	sp3, located after a complete inverted-residual block.
Split reference	Layer-/feature-level partitioning following (Kang et al., 2017; Ko et al., 2018).
CIFAR-10 / MobileNetV2 / sp6	
Dataset	CIFAR-10
Model	MobileNetV2 (Sandler et al., 2018)
Edge-side	`features[0:7]`, ending after a complete inverted-residual block; **55,488 parameters**.
Cloud-side	`features[7:]` followed by global average pooling and the classifier; **2,181,194 parameters**.
Split point	sp6, located after a complete inverted-residual block.
Split reference	Layer-/feature-level partitioning following (Kang et al., 2017; Ko et al., 2018).
CIFAR-100 / ResNet18 / sp2	
Dataset	CIFAR-100
Model	ResNet18 (He et al., 2016)
Edge-side	Stem, `layer1`, and `layer2`; **675,392 parameters**.
Cloud-side	`layer3`, `layer4`, global average pooling, and the 100-class classifier; **10,544,740 parameters**.
Split point	sp2, located after the complete `layer2` residual stage.
Split reference	Stage-level split inference following (Vepakomma et al., 2018; Kang et al., 2017).
CIFAR-100 / ResNet50 / sp2	
Dataset	CIFAR-100
Model	ResNet50 (He et al., 2016)
Edge-side	Stem, `layer1`, and `layer2`; **1,437,248 parameters**.
Cloud-side	`layer3`, `layer4`, global average pooling, and the 100-class classifier; **22,268,004 parameters**.
Split point	sp2, located after the complete `layer2` residual stage.
Split reference	Stage-level partitioning following (Kang et al., 2017; Liu et al., 2024).
CIFAR-100 / VGG16-BN / sp23	
Dataset	CIFAR-100
Model	VGG16-BN (Simonyan & Zisserman, 2014)
Edge-side	`features[0:24]`, including the first three convolutional stages through the third max-pooling layer; **1,737,792 parameters**.
Cloud-side	`features[24:]` followed by adaptive average pooling and the 100-class classifier; **13,036,644 parameters**.
Split point	sp23, located after the third max-pooling stage.
Split reference	Stage-level partitioning following (Liu et al., 2024).
ImageNet-1K / ResNet50 / sp3	
Dataset	ImageNet-1K
Model	ResNet50 (He et al., 2016)
Edge-side	Standard stem, `layer1`, `layer2`, and `layer3`; **8,543,296 parameters**.
Cloud-side	`layer4`, global average pooling, and the 1,000-class classifier; **17,013,736 parameters**.

Continued on next page

Table B.1 – continued from previous page

Item	Configuration
Split point	sp3, located after the complete `layer3` residual stage.
Split reference	Layer-/feature-level partitioning following (Kang et al., 2017; Ko et al., 2018).
Chest CT / ResNet18 / sp2	
Dataset	Chest CT
Model	ResNet18 (He et al., 2016)
Edge-side	Standard stem, `layer1`, and `layer2`; **683,072 parameters**.
Cloud-side	`layer3`, `layer4`, global average pooling, and the 4-class classifier; **10,495,492 parameters**.
Split point	sp2, located after the complete `layer2` residual stage.
Split reference	Stage-level split inference following (Vepakomma et al., 2018; Kang et al., 2017).
UPMC Food-101 / ViLT-B/32 / sp4	
Dataset	UPMC Food-101
Model	ViLT-B/32 (Kim et al., 2021b)
Edge-side	Multimodal embeddings and Transformer blocks 0–3; **54,299,904 parameters**.
Cloud-side	Transformer blocks 4–11, final LayerNorm, pooler, and the 101-class classifier; **57,372,773 parameters**.
Split point	sp4, located after the fourth complete Transformer block.
Split reference	Block-level split inference following (Vepakomma et al., 2018; Kang et al., 2017).

B.3 Training Configurations

Table B.2 summarizes the base training configurations used across different datasets, model architectures, and split points.

Table B.2: **Training configurations across datasets, architectures, and split points.**

Task / Model	Split point / Edge output	Batch	Epochs	Optimizer / Learning rate	Learning-rate schedule
CIFAR-10 / MobileNetV2	sp1: $16 \times 32 \times 32$; sp3: $24 \times 16 \times 16$; sp6: $32 \times 8 \times 8$	128	300	SGD, lr $= 0.1$, momentum $= 0.9$, weight decay $= 5 \times 10^{-4}$	LambdaLR with cosine decay. All methods decay over the first 250 epochs and then remain at the minimum learning rate.
CIFAR-100 / ResNet-18	sp2, after layer2: $128 \times 16 \times 16$	128	200	SGD, lr $= 0.1$, momentum $= 0.9$, Nesterov, weight decay $= 5 \times 10^{-4}$	CosineAnnealingLR, $T_{\max} = 200$, $\eta_{\min} = 10^{-3}$
CIFAR-100 / ResNet-50	sp2, after layer2: $512 \times 16 \times 16$	128	200	SGD, lr $= 0.1$, momentum $= 0.9$, Nesterov, weight decay $= 5 \times 10^{-4}$	CosineAnnealingLR, $T_{\max} = 200$, $\eta_{\min} = 10^{-3}$
CIFAR-100 / VGG-16	sp23, after pool3: $256 \times 4 \times 4$	128	200	SGD, lr $= 0.01$, momentum $= 0.9$, Nesterov, weight decay $= 5 \times 10^{-4}$	CosineAnnealingLR, $T_{\max} = 200$, $\eta_{\min} = 10^{-3}$
ImageNet-1K / ResNet-50	sp3, after layer3: $\sim 1024 \times 14 \times 14$	64	100	SGD, lr $= 0.01$, momentum $= 0.9$, weight decay $= 10^{-4}$	CosineAnnealingLR, $T_{\max} = 100$, $\eta_{\min} = 10^{-5}$
Chest CT / ResNet-18	sp2, after layer2: $128 \times 28 \times 28$	32	100	SGD, lr $= 0.01$, momentum $= 0.9$, Nesterov, weight decay $= 10^{-4}$	CosineAnnealingLR, $T_{\max} = 100$, $\eta_{\min} = 10^{-4}$
UPMC Food-101 / ViLT-B/32	sp4, after the first four Transformer blocks	64	100	AdamW; ViLT backbone: lr $= 5 \times 10^{-6}$; classification head: lr $= 2 \times 10^{-5}$; transmission module: lr $= 10^{-4}$; weight decay $= 0.01$	CosineAnnealingLR, $T_{\max} = 100$, $\eta_{\min} = 10^{-7}$

C Communication Perturbation Modeling and Analysis

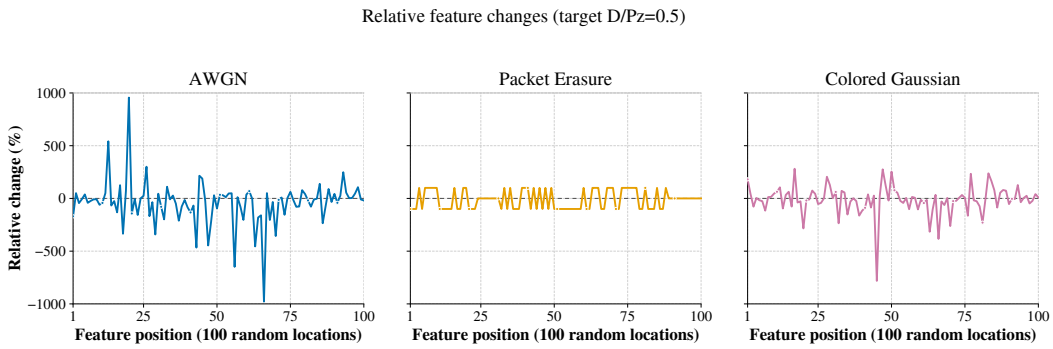


Figure C.1: **Relative intermediate-feature changes under different perturbations.** Relative changes at 100 randomly sampled feature locations under AWGN, packet erasure, and colored Gaussian noise, with the normalized distortion target fixed at $D/P_z = 0.5$. The three perturbations produce clearly different feature-level variation patterns, reflecting their distinct effects on intermediate representations.

AWGN. AWGN represents dense, independent, zero-mean additive perturbations, such as thermal and receiver noise. The noise variance is determined by the instantaneous feature power P_z and the prescribed SNR. We evaluate $\mathrm{SNR} \in \{-10, -5, 0, 5, 10\}$ dB:

$$y = z + n, \qquad n \sim \mathcal{N}(0, \sigma^2 I), \qquad \sigma^2 = \frac{P_z}{10^{\mathrm{SNR}/10}}. \tag{C.1}$$

Since the noise samples are independently drawn across feature locations, AWGN produces dense but spatially uncorrelated feature perturbations. As illustrated in Fig. C.1, the relative changes at randomly sampled feature locations exhibit irregular positive and negative variations over a broad range.

Packet Erasure. Packet erasure represents structured information loss caused by unsuccessful packet transmission. We flatten the intermediate feature and divide it into packets of length 256, where each packet is independently erased with probability p. We evaluate $p \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$:

$$y_k = m_k z_k, \qquad m_k \sim \mathrm{Bernoulli}(1-p), \tag{C.2}$$

where z_k and y_k denote the k-th transmitted and received packets, respectively. Unlike additive noise, packet erasure removes entire groups of feature values, producing discrete and structured information loss. This behavior is also reflected in Fig. C.1, where many sampled locations remain unchanged while erased locations exhibit abrupt feature changes.

Colored Gaussian Noise. Colored Gaussian noise represents spatially correlated additive perturbations for CNN feature maps and locally token-correlated perturbations for ViLT features. Starting from white Gaussian noise u, we apply a convolution kernel K to introduce local correlation and then normalize the resulting perturbation according to the feature power. We evaluate $\alpha \in \{0.5, 1.0, 1.5, 2.0, 2.5\}$:

$$u \sim \mathcal{N}(0, I), \qquad \widetilde{n} = K * u, \qquad n = \alpha \sqrt{P_z} \frac{\widetilde{n}}{\sqrt{P_{\widetilde{n}} + \varepsilon}}, \qquad y = z + n. \tag{C.3}$$

Here, $*$ denotes convolution, P_z is the average power of the transmitted feature, $P_{\widetilde{n}}$ is the empirical average power of the filtered noise, and $\varepsilon = 10^{-12}$ is used for numerical stability. For CNN feature maps, we use the fixed 3×3 kernel

$$K_{\mathrm{CNN}} = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, \tag{C.4}$$

which is applied independently to each channel using depthwise convolution with stride one and padding one. For ViLT, the feature tensor is not reshaped into a two-dimensional image. Instead, we use the one-dimensional kernel

$$K_{\text{token}} = \frac{1}{4}[1, 2, 1], \tag{C.5}$$

applied along the concatenated token dimension independently for each hidden dimension. Thus, neighboring perturbations are statistically correlated through K, yielding a structured additive-noise pattern. In contrast to AWGN, the resulting feature-level variations are visibly different from both independent Gaussian perturbations and the discrete changes induced by packet erasure, as shown in Fig. C.1.

Distributional Relationship Analysis. To further quantify the structural differences among AWGN, packet erasure, and colored Gaussian noise, we characterize each perturbation using several distributional statistics. Given a clean intermediate feature z and its perturbed counterpart y, the induced perturbation is $n = y - z$. We first measure the normalized distortion magnitude as

$$R = \frac{D}{P_z} = \frac{\text{mean}(n^2)}{\text{mean}(z^2)}, \qquad r_{\text{RMS}} = \sqrt{R}. \tag{C.6}$$

To characterize spatial and distributional structures, we compute the horizontal and vertical correlations and the excess kurtosis:

$$\rho_h = \text{Corr}(n_{h,w}, n_{h,w+1}), \qquad \rho_v = \text{Corr}(n_{h,w}, n_{h+1,w}), \tag{C.7}$$

$$\kappa_e = \frac{\mathbb{E}[(n - \mu_n)^4]}{\text{Var}(n)^2} - 3. \tag{C.8}$$

For packet erasure, q denotes the packet-loss ratio, while $q = 0$ for the two additive perturbations. Each perturbation is then represented by

$$\mathbf{v} = [q, \rho_h, \rho_v, \kappa_e, \sqrt{D/P_z}]. \tag{C.9}$$

The structural similarity between perturbations i and j is measured using cosine similarity and the corresponding angular distance:

$$S_{ij} = \frac{\mathbf{v}_i^\top \mathbf{v}_j}{\|\mathbf{v}_i\|_2 \|\mathbf{v}_j\|_2}, \qquad \theta_{ij} = \arccos(S_{ij}). \tag{C.10}$$

A smaller S_{ij}, or equivalently a larger θ_{ij}, indicates greater structural dissimilarity between the two perturbation types.

Using 256 test features and 10 independent channel realizations under the matched distortion level $D/P_z = 0.5$, the cosine similarities are 0.1600 for AWGN–packet erasure, 0.5991 for AWGN–colored Gaussian noise, and 0.1582 for packet erasure–colored Gaussian noise. These results confirm that the three perturbations exhibit substantially different statistical structures, with packet erasure being particularly distinct from the two additive perturbations.

Overall, both the feature-level visualization and the quantitative distributional analysis demonstrate clear structural differences among the three perturbations. Fig. C.1 reveals distinct local variation patterns, while the distribution signatures and pairwise similarity analysis further confirm their differences in sparsity, spatial correlation, and distributional characteristics. These complementary observations indicate that AWGN, packet erasure, and colored Gaussian noise represent distinct perturbation families rather than minor variations of the same corruption process, making them well suited for evaluating cross-noise robustness and generalization to unmatched channel perturbations.

D Additional Ablation Studies

D.1 Analysis of the Learned Local SNR Distribution

To examine whether FNT learns a selective local noise allocation rather than collapsing to uniform or globally strong perturbations, we analyze the final local-SNR distributions across nine model configurations. We fix the reference SNR to $T = -5$ dB and the local SNR span to $\Delta_S = 4$ dB.

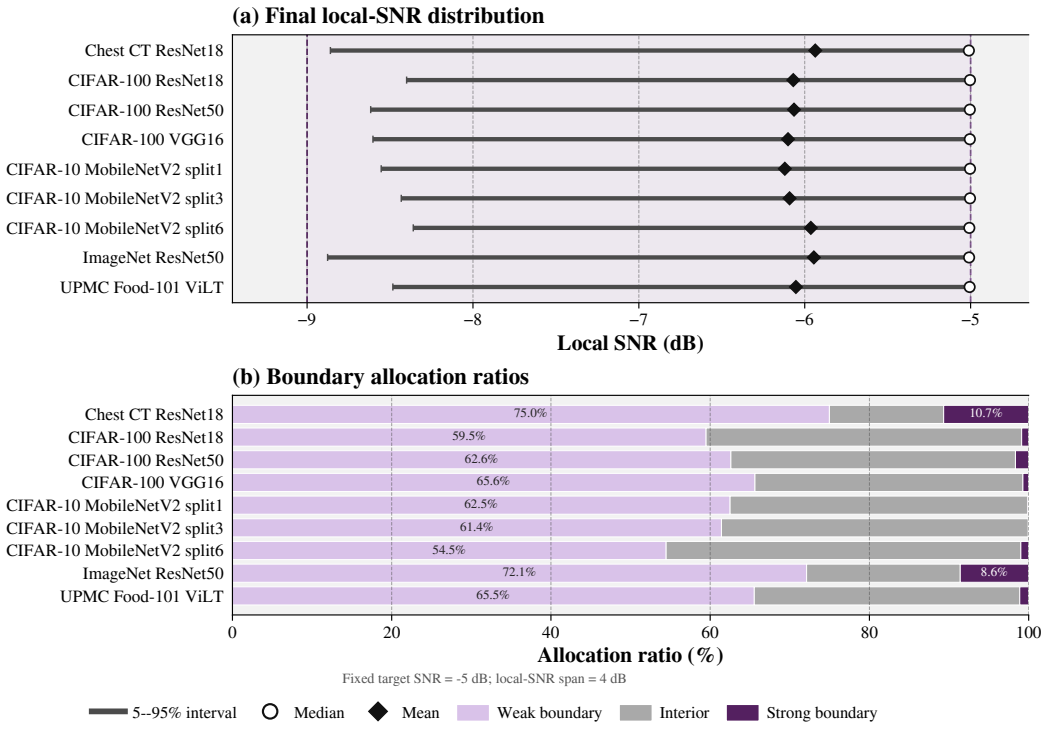


Figure D.1: Final local-SNR distributions and boundary-allocation ratios across nine model configurations. (a) The 5th–95th percentile intervals, means, and medians of the local SNRs. (b) The proportions of feature locations near the weak boundary, within the interior region, and near the strong boundary. The diagnostic uses a fixed target SNR of -5 dB and a local-SNR span of 4 dB.

According to $\mathrm{SNR}_i = T - \Delta_S a_i$ with $a_i \in [0, 1]$, all local SNRs are therefore bounded within $[-9, -5]$ dB.

As shown in Table D.1 and Fig. D.1(a), the mean local SNRs across different tasks, architectures, and split points remain within a narrow range of approximately -5.94 to -6.12 dB, while the medians are consistently close to the weak-noise boundary of -5 dB. This indicates that most feature locations receive relatively mild perturbations, whereas only a subset is assigned substantially lower SNRs. Table D.2 and Fig. D.1(b) further show that the proportion of locations near the strong-noise boundary is generally small, although the exact allocation ratio varies across model configurations. These results demonstrate that FNT maintains the prescribed SNR bounds while selectively concentrating stronger perturbations on a limited subset of feature locations. Moreover, the similar overall allocation pattern across different tasks, architectures, and split points suggests that this behavior is stable across model configurations rather than being specific to a particular network.

D.2 Effect of Per-Sample Standardization

Per-sample standardization is introduced to remove the scale and offset variations in the sensitivity scores produced by ScoreNet across different models and split points, allowing the subsequent shifted-tanh mapping to operate within a consistent numerical range. As shown in Table D.3, the raw ScoreNet outputs exhibit substantial differences in both mean and variance across configurations. For example, the sensitivity scores are -1.943 ± 1.876 for CIFAR-10 MobileNetV2 sp6, -151.268 ± 104.433 for CIFAR-100 ResNet50, and -10777.387 ± 11087.060 for ImageNet ResNet50.

Directly applying the shifted-tanh mapping

$$r_i = \kappa + \frac{1}{2} \operatorname{arctanh}(2a_i - 1). \tag{D.1}$$

Table D.1: **Distribution of the learned local SNRs.**

Model	Mean	Std.	5th Perc.	Median	95th Perc.
Chest CT ResNet18	-5.937	1.607	-8.859	-5.011	-5.002
CIFAR-100 ResNet18	-6.069	1.378	-8.400	-5.004	-5.002
CIFAR-100 ResNet50	-6.065	1.464	-8.617	-5.005	-5.002
CIFAR-100 VGG16	-6.101	1.531	-8.603	-5.005	-5.003
CIFAR-10 MobileNetV2 sp1	-6.120	1.454	-8.553	-5.004	-5.003
CIFAR-10 MobileNetV2 sp3	-6.091	1.441	-8.432	-5.004	-5.002
CIFAR-10 MobileNetV2 sp6	-5.963	1.224	-8.360	-5.008	-5.001
ImageNet ResNet50	-5.945	1.562	-8.876	-5.009	-5.002
UPMC Food-101 ViLT	-6.053	1.471	-8.483	-5.006	-5.002

Table D.2: **Proportion of local SNRs near the weak and strong boundaries.**

Model	Near -5 dB Weak Boundary	Near -9 dB Strong Boundary
Chest CT ResNet18	74.97%	10.68%
CIFAR-100 ResNet18	59.49%	0.88%
CIFAR-100 ResNet50	62.60%	1.66%
CIFAR-100 VGG16	65.62%	0.72%
CIFAR-10 MobileNetV2 sp1	62.50%	0.12%
CIFAR-10 MobileNetV2 sp3	61.42%	0.05%
CIFAR-10 MobileNetV2 sp6	54.47%	0.98%
ImageNet ResNet50	72.11%	8.59%
UPMC Food-101 ViLT	65.54%	1.12%

to scores with such different scales would make a fixed κ have inconsistent meanings across model configurations. Large shifts or scales may push the mapping into its saturation regions, causing many feature locations to receive similar allocation weights and reducing the effective differentiation of local SNRs.

After standardization, the score distributions consistently approach

$$\mu \approx 0, \qquad \sigma \approx 1, \tag{D.2}$$

across all evaluated models and split points. Consequently, a fixed mapping shift such as $\kappa = 1$ retains a consistent interpretation across configurations. Since standardization is an affine transformation with a positive scale factor, it preserves the relative ordering of feature locations while removing sample-dependent offset and scale differences. Therefore, FNT performs local SNR allocation according to standardized relative fragility scores rather than the absolute magnitude of the raw ScoreNet outputs.

D.3 Analysis of ScoreNet Capacity and Saturation

To examine whether the robustness gain of FNT depends on a larger ScoreNet capacity or a wider spatial receptive field, we compare the default lightweight ScoreNet with a structurally matched complex variant. Both variants contain four convolutional layers with the same $C \to C$ channel dimensions, ReLU activations after the first three layers, and the same output shape and initialization scheme. The only difference is the convolution kernel: the lightweight ScoreNet uses 1×1 convolutions throughout, yielding a 1×1 receptive field and 1,088 parameters for MobileNetV2 sp1, whereas the complex variant replaces them with 3×3 convolutions with padding 1, expanding the effective receptive field to approximately 9×9 and increasing the parameter count to 9,280 ($8.53\times$).

As shown in Fig. D.2, the increased capacity and spatial receptive field do not lead to consistent robustness improvements. The two variants perform similarly under mild and moderate AWGN conditions, with only small differences, whereas the lightweight ScoreNet becomes more favorable as the perturbation becomes stronger. Under the strongest AWGN condition of -10 dB,

Table D.3: **Sensitivity-score distributions before and after standardization.**

Model	Before Standardization	After Standardization
Chest CT ResNet18	-5.589 ± 4.560	-0.001 ± 1.000
CIFAR-100 ResNet18	-2.266 ± 2.251	0.001 ± 1.001
CIFAR-100 ResNet50	-151.268 ± 104.433	0.003 ± 1.002
CIFAR-100 VGG16	-64.706 ± 59.638	-0.000 ± 1.000
CIFAR-10 MobileNetV2 sp1	-2.744 ± 2.154	-0.000 ± 1.000
CIFAR-10 MobileNetV2 sp3	-11.816 ± 7.079	0.000 ± 1.000
CIFAR-10 MobileNetV2 sp6	-1.943 ± 1.876	-0.000 ± 1.000
ImageNet ResNet50	-10777.387 ± 11087.060	0.002 ± 1.001
UPMC Food-101 ViLT	-161.637 ± 138.217	0.000 ± 1.000

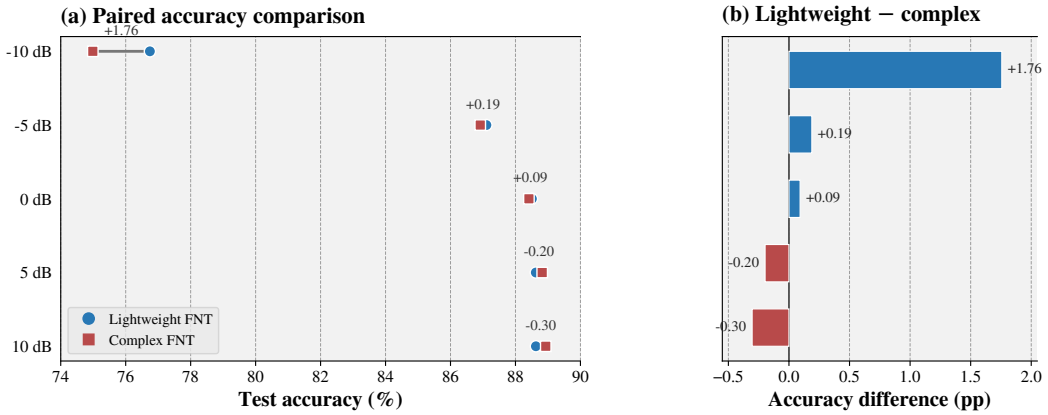


Figure D.2: Comparison between lightweight and complex FNT on CIFAR-10/MobileNetV2 split1 under AWGN. (a) Paired test accuracies across five SNR levels. (b) Accuracy differences between lightweight and complex FNT, computed as Lightweight−Complex.

the lightweight variant outperforms the complex one by 1.76 percentage points, while the complex model shows only marginal advantages under weaker perturbations. These results support the lightweight ScoreNet as an effective choice for fragility-aware training in the evaluated configuration. Increasing its capacity and spatial receptive field does not yield consistent robustness improvements in this comparison.

D.4 Effect of the Clean-Loss Term

The noisy loss provides the robustness-training objective, while the clean loss supplies auxiliary supervision. Adding the clean-loss term yields a smaller observed validation-accuracy decline on UPMC Food-101 and trades some accuracy at -10 dB for gains at higher SNRs (Tables D.4 and D.5).

Table D.4: UPMC Food-101 training diagnostics. Arrows indicate earlier $\rightarrow$ later values.

Variant	Validation accuracy (%)	Noisy training loss
Full FNT	$92.27 \rightarrow 91.26$	$4.47 \rightarrow 0.80$
w/o clean loss	$90.80 \rightarrow 88.24$	$4.25 \rightarrow 0.60$

Table D.5: Accuracy change (pp): full FNT minus FNT w/o clean loss. Column headers denote SNR (dB).

Model	-10	-5	0	5	10
CIFAR-10 split1	-2.28	$+1.35$	$+1.51$	$+1.56$	$+0.54$
UPMC ViLT	-3.86	$+0.24$	$+1.84$	$+2.45$	$+1.58$

E STATISTICAL RELIABILITY AND ADDITIONAL RESULTS

Table E.1: **Additional evaluation metrics under AWGN.**

ImageNet-1K ResNet50 split3 (Top-5 Accuracy)

Condition	Van.	COM	FGM	DeepJSCC	FNT
Clean	93.96	93.61	94.02	91.90	92.92
-10 dB	12.62	83.30	16.93	44.08	89.23
-5 dB	81.45	93.07	84.66	88.10	92.59
0 dB	92.75	93.62	92.96	91.65	92.96
5 dB	93.72	93.63	93.77	91.93	92.96
10 dB	93.91	93.61	93.93	91.90	92.92

Chest CT ResNet18 split2 (Macro-F1)

Condition	Van.	COM	FGM	DeepJSCC	FNT
Clean	67.82	63.91	66.12	78.71	71.88
-10 dB	7.32	7.32	12.38	12.65	70.38
-5 dB	10.80	22.90	37.54	32.96	75.16
0 dB	49.07	58.19	60.30	68.67	73.67
5 dB	62.59	62.93	64.57	75.77	72.63
10 dB	66.50	63.58	65.76	77.53	72.14

E.1 ADDITIONAL EVALUATION METRICS

To provide a more comprehensive evaluation beyond the primary Top-1 accuracy, we additionally report Top-5 accuracy on ImageNet-1K and Macro-F1 on Chest CT in Table E.1. ImageNet-1K contains 1,000 categories, and Top-5 accuracy provides a complementary measure of whether the correct class remains among the model's highest-ranked predictions under transmission noise. For the four-class Chest CT task, Macro-F1 assigns equal importance to each class and therefore complements overall accuracy by reflecting class-wise precision and recall performance.

The additional metrics exhibit trends consistent with the main results. On ImageNet-1K, FNT maintains a Top-5 accuracy of 89.23% at -10 dB, compared with 83.30% for the strongest baseline, while remaining competitive under milder perturbations. On Chest CT, FNT preserves a Macro-F1 of 70.38% and 75.16% at -10 and -5 dB, respectively, whereas the competing methods degrade substantially under the same conditions. These results indicate that the robustness advantage of FNT is not specific to a single evaluation metric.

E.2 ROBUSTNESS ACROSS TRAINING AND CHANNEL SEEDS

To evaluate the statistical reliability of the robustness results, we examine two independent sources of randomness: training seeds and channel seeds. Different training seeds capture variations caused by model initialization and stochastic optimization, whereas different channel seeds capture variations in the sampled channel perturbations during evaluation.

As shown in Figs. E.2 and E.1, the overall robustness trends remain consistent across both types of random seeds. In particular, FNT consistently maintains high accuracy across the evaluated AWGN SNR range and shows only minor variations under different training or channel realizations. The relative performance trends among the compared methods are also largely preserved. These results indicate that the observed robustness improvements are not attributable to a particular training initialization or channel realization, supporting the statistical reliability of the main findings.

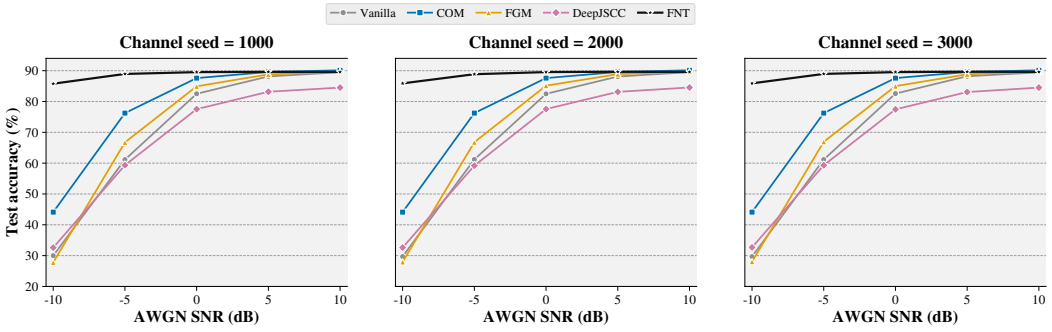


Figure E.1: AWGN test-accuracy curves across three channel seeds. The subplots correspond to channel seeds 1000, 2000, and 3000 with the training seed fixed to 456.

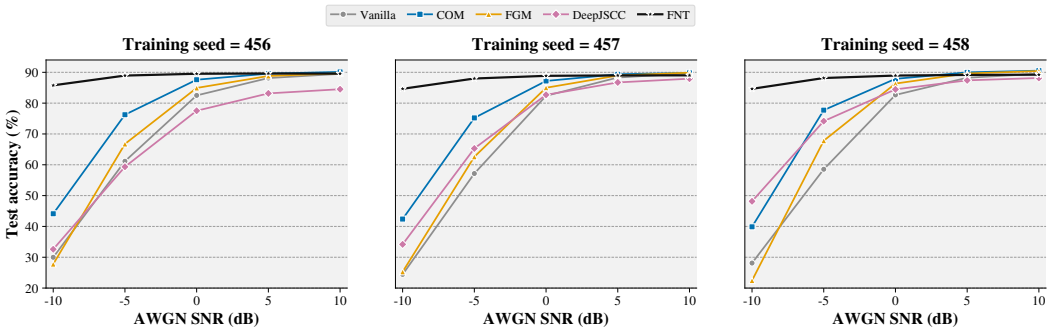


Figure E.2: AWGN test-accuracy curves of Vanilla, COM, FGM, DeepJSCC, and FNT across three training seeds. The three subplots correspond to training seeds 456, 457, and 458, respectively.

Table E.2: **Clean and noisy test accuracy (%) under packet erasure and colored Gaussian noise.** Noisy results are reported as mean ± standard deviation over 10 channel trials.

Packet Erasure

UPMC Food-101 ViLT split4

p	Van.	COM	FGM	DeepJSCC	FNT
Clean	92.89	90.09	92.90	88.94	91.71
0.1	91.56 ± 0.08	88.86 ± 0.06	91.77 ± 0.08	77.96 ± 0.16	90.88 ± 0.10
0.2	87.81 ± 0.22	86.08 ± 0.15	88.05 ± 0.19	55.87 ± 0.27	89.05 ± 0.15
0.3	78.62 ± 0.35	80.63 ± 0.14	79.30 ± 0.27	31.97 ± 0.16	85.70 ± 0.16
0.4	59.78 ± 0.27	70.35 ± 0.26	61.31 ± 0.28	16.89 ± 0.24	79.67 ± 0.23
0.5	31.82 ± 0.26	53.60 ± 0.27	32.92 ± 0.15	8.65 ± 0.13	71.62 ± 0.16

CIFAR-10 MobileNetV2 split6

p	Van.	COM	FGM	DeepJSCC	FNT
Clean	90.02	90.27	90.12	85.06	89.56
0.1	86.53 ± 0.16	88.29 ± 0.21	87.40 ± 0.19	75.83 ± 0.20	88.79 ± 0.17
0.2	81.46 ± 0.25	85.14 ± 0.25	83.13 ± 0.24	67.26 ± 0.37	87.85 ± 0.18
0.3	74.80 ± 0.37	80.20 ± 0.33	77.35 ± 0.23	58.31 ± 0.32	86.01 ± 0.27
0.4	66.46 ± 0.27	73.58 ± 0.36	69.51 ± 0.48	50.49 ± 0.37	83.22 ± 0.21
0.5	56.24 ± 0.56	64.45 ± 0.44	59.96 ± 0.50	42.80 ± 0.35	78.62 ± 0.43

Colored Gaussian Noise

UPMC Food-101 ViLT split4

Scale	Van.	COM	FGM	DeepJSCC	FNT
Clean	92.89	90.09	92.90	88.94	91.71
0.5	92.09 ± 0.07	90.70 ± 0.09	92.12 ± 0.05	88.30 ± 0.10	90.90 ± 0.09
1.0	67.41 ± 0.22	91.55 ± 0.08	67.61 ± 0.19	73.89 ± 0.14	88.51 ± 0.11
1.5	9.79 ± 0.13	90.67 ± 0.08	9.73 ± 0.15	34.61 ± 0.28	86.76 ± 0.06
2.0	2.54 ± 0.08	81.74 ± 0.16	2.66 ± 0.11	10.46 ± 0.21	85.65 ± 0.10
2.5	2.14 ± 0.07	44.57 ± 0.21	2.15 ± 0.08	3.45 ± 0.13	84.56 ± 0.05

CIFAR-10 MobileNetV2 split6

Scale	Van.	COM	FGM	DeepJSCC	FNT
Clean	90.02	90.27	90.12	85.06	89.56
0.5	84.49 ± 0.29	88.06 ± 0.18	86.70 ± 0.24	78.94 ± 0.26	89.33 ± 0.10
1.0	66.22 ± 0.53	79.36 ± 0.47	72.16 ± 0.30	61.08 ± 0.38	88.09 ± 0.14
1.5	44.44 ± 0.56	63.15 ± 0.32	51.34 ± 0.33	42.55 ± 0.38	85.72 ± 0.14
2.0	30.06 ± 0.43	47.34 ± 0.50	35.45 ± 0.69	31.14 ± 0.41	81.77 ± 0.27
2.5	22.61 ± 0.24	36.06 ± 0.63	26.03 ± 0.32	24.21 ± 0.48	76.63 ± 0.22

E.2.1 Training Convergence and Optimization Stability

To verify the convergence and optimization stability of FNT, Fig. E.3 shows the training and clean validation curves of all compared methods on CIFAR-100 with ResNet18 split2. Although FNT introduces an additional adversarial update of ScoreNet, its optimization remains stable throughout training. The training accuracy gradually approaches saturation, while the training loss continuously decreases and converges. Meanwhile, the clean validation accuracy and loss also stabilize as training proceeds, indicating that the additional fragility-aware optimization does not prevent the split model from reaching a stable solution.

For a fair comparison, all competing methods are trained following the same training protocol and convergence criterion, rather than being evaluated from partially optimized checkpoints. As shown in Fig. E.3, all methods exhibit stable optimization behavior under this setting. These results confirm that the robustness improvements of FNT are obtained from a properly converged model rather than from unstable or incomplete training.

E.2.2 Detailed Results under Different Noise Strengths

For completeness and to facilitate more detailed quantitative comparisons, we report the full numerical results under different perturbation strengths in Tables E.3 and E.2. These tables provide the mean test accuracy and standard deviation over 10 channel trials at each evaluated AWGN SNR, packet-erasure probability, and colored-Gaussian noise scale, complementing the robustness curves presented in the main text.

Overall, the detailed results show a consistent trend: as the perturbation becomes stronger, the performance degradation of conventional methods becomes increasingly pronounced, whereas FNT

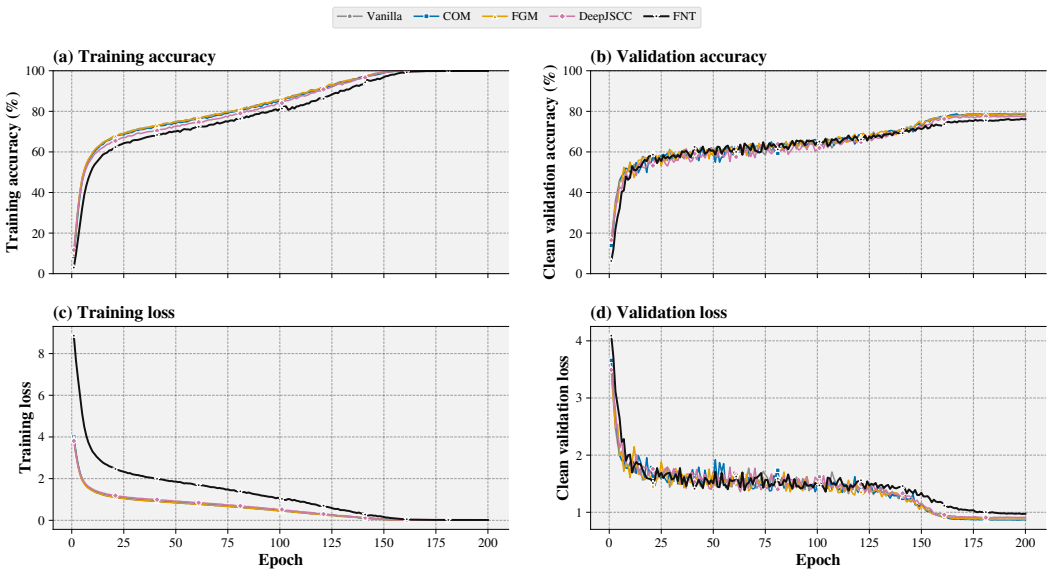


Figure E.3: Training convergence of five methods on CIFAR-100 with ResNet18 split2. The four panels show training accuracy, clean validation accuracy, training loss, and clean validation loss, respectively.

maintains substantially higher accuracy in most severe-noise settings. The advantage is particularly evident at low AWGN SNRs, high packet-erasure probabilities, and large colored-Gaussian noise scales. Under relatively mild perturbations, the performance gap becomes smaller and some baselines achieve comparable or slightly higher accuracy. Across repeated channel realizations, the standard deviations are generally small, indicating that the observed trends are stable. Notably, FNT can occasionally exhibit a slightly larger standard deviation than some baselines under the strongest perturbations. This should not be interpreted as lower robustness: in several such cases, the baseline accuracy has already collapsed toward a near-chance performance floor, where consistently uninformative predictions naturally leave little room for accuracy variation across channel realizations. In contrast, FNT remains in a non-degenerate operating regime with substantially higher accuracy, so variations among severe channel realizations can still be reflected in its performance. Therefore, the standard deviations should be interpreted jointly with the corresponding mean accuracy rather than as an independent measure of robustness. Overall, these results indicate that the main benefit of FNT lies in preserving reliable task performance under challenging transmission conditions.

Table E.3: **Clean and AWGN test accuracy (%).** AWGN results are reported as mean ± standard deviation over 10 channel trials.

CIFAR-10 MobileNetV2 split1

SNR	Van.	COM	FGM	DeepJSCC	FNT
Clean	90.02	89.85	90.56	85.66	88.55
-10	18.82 ± 0.26	16.71 ± 0.18	12.33 ± 0.18	12.22 ± 0.12	76.75 ± 0.21
-5	43.38 ± 0.38	66.81 ± 0.16	40.96 ± 0.20	25.54 ± 0.19	87.11 ± 0.10
0	75.19 ± 0.23	86.63 ± 0.19	79.21 ± 0.16	61.81 ± 0.26	88.51 ± 0.11
5	86.93 ± 0.20	89.66 ± 0.12	88.36 ± 0.17	82.82 ± 0.15	88.63 ± 0.10
10	89.37 ± 0.13	89.92 ± 0.09	90.02 ± 0.06	85.81 ± 0.17	88.63 ± 0.07

CIFAR-10 MobileNetV2 split3

SNR	Van.	COM	FGM	DeepJSCC	FNT
Clean	90.02	89.56	90.42	80.91	87.10
-10	15.68 ± 0.15	20.96 ± 0.17	22.02 ± 0.34	17.72 ± 0.29	78.53 ± 0.22
-5	23.93 ± 0.23	58.44 ± 0.30	60.61 ± 0.40	33.76 ± 0.18	86.18 ± 0.32
0	73.27 ± 0.23	85.94 ± 0.08	84.83 ± 0.29	70.86 ± 0.40	87.01 ± 0.07
5	87.86 ± 0.19	89.03 ± 0.17	89.17 ± 0.12	80.99 ± 0.19	87.01 ± 0.16
10	89.61 ± 0.15	89.50 ± 0.09	90.12 ± 0.11	81.22 ± 0.19	87.06 ± 0.09

CIFAR-10 MobileNetV2 split6

SNR	Van.	COM	FGM	DeepJSCC	FNT
Clean	90.02	90.27	90.12	85.06	89.56
-10	29.68 ± 0.30	44.11 ± 0.45	28.17 ± 0.52	32.94 ± 0.28	85.81 ± 0.32
-5	61.20 ± 0.41	76.24 ± 0.21	66.86 ± 0.36	59.17 ± 0.23	88.97 ± 0.09
0	82.42 ± 0.28	87.60 ± 0.23	85.07 ± 0.14	77.44 ± 0.15	89.58 ± 0.15
5	88.05 ± 0.18	89.57 ± 0.16	88.85 ± 0.24	83.19 ± 0.28	89.66 ± 0.09
10	89.45 ± 0.11	90.13 ± 0.11	89.66 ± 0.13	84.57 ± 0.17	89.60 ± 0.05

CIFAR-100 ResNet18 split2

SNR	Van.	COM	FGM	DeepJSCC	FNT
Clean	78.06	77.44	77.91	66.90	75.28
-10	39.45 ± 0.41	55.51 ± 0.30	51.06 ± 0.19	27.79 ± 0.29	72.44 ± 0.22
-5	69.41 ± 0.31	73.08 ± 0.20	72.58 ± 0.31	62.02 ± 0.30	74.75 ± 0.07
0	76.03 ± 0.15	76.47 ± 0.10	76.56 ± 0.13	68.40 ± 0.26	75.05 ± 0.10
5	77.42 ± 0.11	77.12 ± 0.13	77.45 ± 0.16	68.13 ± 0.11	75.18 ± 0.11
10	77.84 ± 0.18	77.31 ± 0.13	77.75 ± 0.08	67.49 ± 0.14	75.31 ± 0.07

Table E.3: **Clean and AWGN test accuracy (%) (continued).** AWGN results are reported as mean ± standard deviation over 10 channel trials.

CIFAR-100 ResNet50 split2

SNR	Van.	COM	FGM	DeepJSCC	FNT
Clean	77.95	77.65	77.72	71.64	76.39
-10	56.28 ± 0.34	68.25 ± 0.36	69.13 ± 0.20	45.09 ± 0.32	75.11 ± 0.23
-5	74.08 ± 0.18	76.08 ± 0.23	76.40 ± 0.22	67.68 ± 0.27	76.12 ± 0.14
0	76.99 ± 0.22	77.29 ± 0.14	77.48 ± 0.21	71.09 ± 0.18	76.31 ± 0.11
5	77.46 ± 0.14	77.55 ± 0.09	77.68 ± 0.07	71.54 ± 0.13	76.30 ± 0.08
10	77.78 ± 0.07	77.64 ± 0.08	77.68 ± 0.05	71.66 ± 0.11	76.32 ± 0.07

CIFAR-100 VGG16 split23

SNR	Van.	COM	FGM	DeepJSCC	FNT
Clean	70.18	69.95	70.61	68.30	68.83
-10	30.45 ± 0.33	44.91 ± 0.41	34.96 ± 0.24	27.63 ± 0.49	61.71 ± 0.27
-5	56.36 ± 0.21	63.30 ± 0.38	59.23 ± 0.30	52.78 ± 0.37	67.25 ± 0.12
0	65.93 ± 0.27	68.34 ± 0.20	67.25 ± 0.19	63.67 ± 0.24	68.40 ± 0.16
5	68.80 ± 0.20	69.47 ± 0.15	69.58 ± 0.25	67.11 ± 0.20	68.62 ± 0.10
10	69.72 ± 0.15	69.79 ± 0.18	70.20 ± 0.08	67.91 ± 0.19	68.75 ± 0.10

ImageNet-1K ResNet50 split3

SNR	Van.	COM	FGM	DeepJSCC	FNT
Clean	77.86	76.86	77.87	73.64	75.83
-10	4.96 ± 0.09	59.00 ± 0.16	6.99 ± 0.11	20.54 ± 0.11	68.76 ± 0.08
-5	57.31 ± 0.15	75.88 ± 0.08	61.32 ± 0.12	66.22 ± 0.15	75.09 ± 0.06
0	75.61 ± 0.06	76.88 ± 0.03	75.80 ± 0.10	73.09 ± 0.06	75.78 ± 0.05
5	77.39 ± 0.05	76.91 ± 0.06	77.45 ± 0.06	73.66 ± 0.07	75.83 ± 0.03
10	77.74 ± 0.06	76.91 ± 0.05	77.73 ± 0.02	73.64 ± 0.04	75.83 ± 0.04

Chest CT ResNet18 split2

SNR	Van.	COM	FGM	DeepJSCC	FNT
Clean	66.03	60.95	63.17	77.46	68.25
-10	17.14 ± 0.00	17.14 ± 0.00	20.51 ± 0.98	20.98 ± 1.05	66.44 ± 1.63
-5	19.75 ± 0.65	28.25 ± 0.85	46.13 ± 2.20	38.35 ± 2.15	71.49 ± 1.30
0	52.86 ± 2.32	56.89 ± 1.20	59.37 ± 1.46	67.43 ± 1.10	70.06 ± 0.40
5	62.44 ± 1.13	59.81 ± 1.12	61.81 ± 0.62	74.57 ± 1.09	69.05 ± 0.85
10	64.98 ± 1.45	60.54 ± 0.82	62.76 ± 0.91	76.32 ± 0.60	68.51 ± 0.47

UPMC Food-101 ViLT split4

SNR	Van.	COM	FGM	DeepJSCC	FNT
Clean	92.89	90.09	92.90	88.94	91.71
-10	1.92 ± 0.10	7.60 ± 0.17	1.90 ± 0.07	1.79 ± 0.05	82.97 ± 0.11
-5	3.47 ± 0.06	88.35 ± 0.10	3.47 ± 0.07	20.71 ± 0.23	86.33 ± 0.10
0	67.45 ± 0.21	91.65 ± 0.04	67.82 ± 0.28	78.44 ± 0.14	88.56 ± 0.09
5	91.70 ± 0.06	90.84 ± 0.07	91.74 ± 0.10	88.28 ± 0.07	90.69 ± 0.06
10	92.68 ± 0.08	90.38 ± 0.04	92.75 ± 0.04	88.83 ± 0.09	91.43 ± 0.07

Table E.4: **Cross-sample correlation and Top-10% overlap of the learned noise-allocation maps.**

Model	Cross-sample correlation	Top-10% overlap
Chest CT ResNet-18	0.9995	0.5539
CIFAR-100 VGG-16	0.9948	0.5834
CIFAR-10 MobileNetV2 Split1	0.9997	0.7253
CIFAR-10 MobileNetV2 Split3	0.9012	0.6008

E.3 Analysis of Learned Noise-Allocation Patterns

To characterize the noise-allocation patterns learned by FNT, we analyze the allocation maps across different input samples using two statistics: cross-sample correlation and Top-10% overlap. Cross-sample correlation measures the similarity between allocation maps after removing global mean and scale, while Top-10% overlap measures the fraction of highly allocated feature locations shared across samples.

As shown in Table E.4, the learned allocation maps exhibit substantial cross-sample consistency across the evaluated configurations. The cross-sample correlations range from 0.9012 to 0.9997, while the Top-10% overlap varies across models and split points. These results suggest that the learned allocation maps contain a considerable amount of shared structure across samples, with the detailed allocation patterns depending on the model configuration and split point. This analysis provides a descriptive view of the spatial structure learned by the noise-allocation mechanism.

Intervention-Based Assessment of Fragility Ranking. To directly examine whether the fragility scores learned by FNT correspond to task-sensitive feature locations, we perform a controlled perturbation intervention on CIFAR-10 with MobileNetV2 split6. For each of 256 test samples, feature locations are ranked according to their ScoreNet scores, and perturbations are applied to either the Top-10% highest-scoring locations, a randomly selected 10% of locations, or the Bottom-10% lowest-scoring locations. All three groups use exactly the same per-sample L_2 perturbation budget and Gaussian perturbation directions, so that the only difference lies in the selected feature locations. We evaluate two perturbation budgets equivalent to 0 dB and -5 dB noise levels. Each intervention is repeated 10 times per image to reduce randomness from perturbation sampling, while statistical analysis is conducted at the image level to avoid treating multiple perturbations of the same image as independent samples. We compare the resulting increase in task loss and decrease in classification accuracy to assess whether higher ScoreNet rankings correspond to greater feature fragility.

Table E.5: **Intervention-based assessment of the learned fragility ranking.** The same per-sample L_2 perturbation budget is applied to the Top-10%, random 10%, or Bottom-10% feature locations ranked by ScoreNet. Loss statistics are computed at the image level over 256 test samples. The 95% confidence intervals are computed by bootstrapping over the 256 image-level observations.

Budget	Perturbed locations	Mean loss increase	95% CI	Accuracy drop (pp)
0 dB	Top-10% ScoreNet	0.017567	$[-0.002773, 0.036888]$	1.680
	Random-10%	0.002453	$[-0.011992, 0.015631]$	0.977
	Bottom-10% ScoreNet	-0.003346	$[-0.013066, 0.005450]$	0.156
-5 dB	Top-10% ScoreNet	0.069644	$[0.028460, 0.108527]$	3.984
	Random-10%	0.021005	$[-0.010523, 0.049937]$	1.562
	Bottom-10% ScoreNet	0.001061	$[-0.021496, 0.022254]$	0.938

As shown in Table E.5, perturbing the Top-10% locations ranked by ScoreNet yields larger mean loss increases and accuracy drops than perturbing random or Bottom-10% locations under the same perturbation budget. These differences are more pronounced under the stronger -5 dB-equivalent budget. The results provide empirical support for interpreting the ScoreNet ranking as an indicator of relative feature fragility in the evaluated configuration.