APPENDIX: Leveraging Test-Time Consensus Prediction for Robustness against Unseen Noise

In this appendix, we provide details that could not be included in the main paper owing to space constraints, including: (i) results on all 4 kinds of noise on CIFAR10-C dataset such as: Gaussian Noise, Impulse Noise, Shot Noise and Speckle Noise using WRN 40-2 architecture with comparisons to natural/normal training, joint training [11] and test time training [28]; (ii) similar results on MNIST-C dataset using LeNet architecture respectively; (iii) effect of individual parts of our proposed model for CIFAR10-C, MNIST-C and TinyImageNet-C datasets; (iv) results on MNIST and TinyImageNet clean data for test time training method [28] along with individual parts of our proposed method as well as together; (v) more visualizations of effect of quantized latent (VQ-VAE) on mitigating noise for sample images from MNIST-C and TinyImageNet-C datasets; (vi) qualitative results on TTT vs TTCP on improving generalization at inference time for complete MNIST-C, TinyImagent-C and ImageNet-C noisy test data; (vii) results on other corruptions from [10], apart from the 4 kinds of noise mentioned above, for CIFAR10-C dataset; as well as (viii) comparative study of TTCP with the backbone network trained with or without augmentation followed by AugMix [12].

A. Architecture Details

We used a VQ architecture similar to what was used for CIFAR10 dataset (Section 4 in main paper) for our experiments with MNIST and TinyImageNet datasets, shown here. The encoder network consists of 2 convolutional layers followed by 2 residual blocks each of which contains 2 convolutional layers. The decoder follows the same architecture in mirrored fashion. The dimension of the embedding vectors are taken to be 256 and 512 for MNIST and TinyImageNet datasets respectively. We used group norm [35] in between convolutional layers in both encoder and decoder modules to handle single sample reconstruction required for TTCP strategy at inference time.

As explained in Sec 3 of the main paper, a teacher network is introduced to extend the TTCP method to TTCP++. As mentioned in Sec 4, we used ResNet-26 as the student (backbone) network for CIFAR10-C (following [28] and [1]), and WRN 28-10 as the teacher network. Going by popular architectures used on MNIST, LeNet and Resnet-18 [9] architectures were used as student and teacher networks respectively for our experiments on the MNIST-C dataset. For experiments on TinyImageNet-C, ResNet-26 and WRN 28-10 architectures were used as student and teacher networks. ResNet-18 and ResNet-152 networks were used as student and teacher networks for the ImageNet-C dataset.

B. Additional Results

MNIST-C: Tables 14, 15, 16 and 17 report comparisons of natural/normal training, test time training [28] and OUR method (comprising all components) on MNIST-C data with all 4 kinds of noise. The results corroborate our claims and show improvement over baseline methods.

CIFAR10-C: In addition to the results on CIFAR10-C in Sec 4 with ResNet-26 and WRN 28-10 as student and teacher networks respectively, we also studied the performance on CIFAR10C with WRN 40-2 as the student (backbone) architecture (since joint training (JT) [11] uses this architecture). We present results on the 4 kinds of noise: Gaussian, Impulse, Shot and Speckle noise on CIFAR10-C using WRN 40-2 and WRN 28-10 architectures as student and teacher networks respectively in Tables 20, 21, 22 and 23. The values reported for natural training and JT [11] are obtained using batch-normalization [15] (as in their work), whereas results for test time training and our method followed group normalization [35]. We included results of test time training (TTT) [28] on the WRN 40-2 architecture for completeness of analysis (note that [28] reported their results on only ResNet-26 architecture).

Performance on clean data: In continuation to the results presented in Table 10 in the main paper for CIFAR10, we show results on MNIST and TinyImageNet clean data using different components of our method, as well as vanilla test time training (TTT) [28] in Tables 24 and 25 respectively. These results show that our method maintains performance on clean data (in fact, improves performance in certain cases) while providing strong performance on unseen noise.

C. More Ablation Studies

Effect of individual parts of our method for CIFAR10-C, MNIST-C and TinyImageNet-C: In Table 13, we presented results for different components of our framework as an average across the severity levels (due to space constraints). We herein present the results for each severity level in Tables 26, 27, 28 and 29. Similar results are presented in Tables 30,31,32,33 and tables 34,35,36,37 for MNIST-C and TinyImageNet-C datasets respectively. Note that across these results, as the severity level increases, the VQ and KD module add value, demonstrating the need for the TTCP++ framework.

Performance against other corruptions: We show the effect of our methods, TTCP and TTCP++, over TTT against other corruptions in [10] for CIFAR10-C and ImageNet-C in Tables 38 and 39. Note that our methods consistently outperform other baselines on all these corruptions too. In particular, both our methods improve performance over baselines on weather-based corruptions consistently. For corruptions based on camera capture such as zoom and contrast, the VQ-VAE module is not ideal for retrieving the

Method	Natural	Baseline TTT [28]	OURS TTCP	OURS TTCP++
SL 1	90.91	96.72	97.08	**97.18**
SL 2	86.55	92.59	93.70	**94.67**
SL 3	76.05	86.72	88.81	**93.92**
SL 4	70.23	80.11	83.97	**91.11**
SL 5	64.65	75.26	79.21	**89.26**
Avg.	77.67	86.28	88.55	**93.22**

Table 14. Results on Gaussian noise

Method	Natural	Baseline TTT [28]	OURS TTCP	OURS TTCP++
SL 1	89.04	96.59	96.61	**96.65**
SL 2	86.21	91.10	92.54	**95.09**
SL 3	82.23	87.12	90.05	**93.17**
SL 4	76.01	82.17	86.72	**91.46**
SL 5	71.43	78.25	83.28	**90.52**
Avg.	80.98	87.04	89.84	**93.38**

Table 15. Results on Impulse noise

Method	Natural	Baseline TTT [28]	OURS TTCP	OURS TTCP++
SL 1	81.93	96.19	96.23	**96.71**
SL 2	80.43	94.72	95.24	**95.81**
SL 3	77.84	88.34	90.35	**94.57**
SL 4	74.82	81.28	84.70	**93.98**
SL 5	69.71	76.91	79.56	**92.69**
Avg.	76.94	87.48	89.21	**94.75**

Table 16. Results on Shot noise

Method	Natural	Baseline TTT [28]	OURS TTCP	OURS TTCP++
SL 1	87.24	96.34	96.71	**96.79**
SL 2	84.18	92.51	93.78	**94.98**
SL 3	81.87	87.22	89.09	**93.80**
SL 4	76.59	81.14	87.16	**91.89**
SL 5	70.02	75.73	81.59	**91.38**
Avg.	79.98	86.59	89.66	**93.76**

Table 17. Results on Speckle noise

Accuracy results of our methods, TTCP and TTCP++, as well as the baseline method, TTT, with Gaussian, Impulse, Shot and Speckle noise on MNIST-C dataset using LeNet and ResNet-18 architectures as student (backbone) and teacher networks respectively. This results show significant improvement of our method over previous method i.e. TTT. TTT = Test time training [28]

latent data manifold (our TTCP module outperforms baselines in these cases however). Exploring other options to retrieve the true data manifold for such camera capture-based corruptions would be an interesting direction of future work.

TTCP vs TTT on MNIST-C, TinyImageNet-C and ImageNet-C: In addition to the results on CIFAR10-C (presented in the main paper in Sec 5), we show significant improvement using TTCP over TTT for MNIST-C, TinyImageNet-C and ImageNet-C datasets in Table 18.

Method	MNIST-C TTT [28]	MNIST-C TTCP	Tiny-ImageNet-C TTT [28]	Tiny-ImageNet-C TTCP	ImageNet-C TTT [28]	ImageNet-C TTCP
Gauss	86.28	**88.55**	40.18	**42.79**	3.1	**3.8**
Impulse	87.04	**89.84**	42.12	**44.49**	3.5	**4.1**
Shot	87.48	**89.21**	43.60	**45.79**	4.5	**5.2**
Speckle	86.59	**89.66**	41.62	**44.78**	NA	NA
Clean	99.57	**99.78**	72.11	**72.65**	69.0	**69.7**

Table 18. Comparative effect of TTT and TTCP on improving generalization at inference time for complete noisy test data from MNIST-C, TinyImageNet-C (accuracy results averaged across 5 severity levels) and ImageNet-C datasets (accuracy results shown for only severity level 5) using LeNet, ResNet-26 and ResNet-18 architectures as backbone networks respectively.

Combining TTCP with AugMix [12]: Considering AugMix [12] performed the best among the augmentation methods (in our results in Sec 4), we studied the possibility of combining our method with AugMix by using AugMix during training, and TTCP at inference. Table 19 shows these results on CIFAR10-C and indicates that our method can give further boost in performance when combined with augmentation methods. Exploring this combination further may be another promising direction of future work to obtain models robust to unseen noise.

Method	TTCP	AugMix [12]	TTCP+AugMix
Gauss	73.84	81.00	**83.29**
Impulse	75.76	86.00	**87.46**
Shot	78.17	85.00	**86.68**
Speckle	77.68	78.00	**80.74**

Table 19. Comparative effect of AugMix [12] and TTCP over both of them individually for complete noisy test data from CIFAR10-C using WRN 40-2 architecture. (accuracy results averaged across 5 severity levels.)

Visualizing effect of quantized latents:: We explained the importance of quantized latents in mitigating unseen noise in input image with visualizations for CIFAR10-C in Sec 3 in the main paper. Here we present more such visualizations for sample images from MNIST-C (Figure 5) and TinyImageNet-C (Figure 6) to show the effect across datasets. It is observed from the visualizations that quantization helps in removing noise at different severity levels (even with severity level 5), and thus reconstructing noise-free images that can be passed on to the next stage of our pipeline in TTCP++.

Method	Natural	Baselines		OURS	
		JT [11]	TTT [28]	TTCP	TTCP++
SL 1	NA	NA	84.19	86.59	**87.23**
SL 2	NA	NA	74.62	78.92	**84.79**
SL 3	NA	NA	66.09	73.38	**82.91**
SL 4	NA	NA	61.34	68.09	**81.04**
SL 5	NA	NA	55.86	62.25	**75.46**
Avg	42.00	52.00	68.42	73.84	**82.29**

Table 20. Results with Gaussian noise (SL = Severity Level)

Method	Natural	Baselines		OURS	
		JT [11]	TTT [28]	TTCP	TTCP++
SL 1	NA	NA	86.21	87.14	**87.96**
SL 2	NA	NA	80.14	82.61	**85.69**
SL 3	NA	NA	74.38	78.91	**84.57**
SL 4	NA	NA	63.81	70.03	**80.71**
SL 5	NA	NA	53.26	60.11	**70.36**
Avg	55.00	61.00	71.56	75.76	**81.86**

Table 21. Results with Impulse noise (SL = Severity Level)

Method	Natural	Baselines		OURS	
		JT [11]	TTT [28]	TTCP	TTCP++
SL 1	NA	NA	87.35	88.12	**89.97**
SL 2	NA	NA	82.53	84.53	**86.58**
SL 3	NA	NA	71.82	77.68	**83.04**
SL 4	NA	NA	68.14	74.19	**78.31**
SL 5	NA	NA	61.74	66.37	**70.19**
Avg	53.00	63.00	74.32	78.17	**81.62**

Table 22. Results with Shot noise (SL = Severity Level)

Method	Natural	Baselines		OURS	
		JT [11]	TTT [28]	TTCP	TTCP++
SL 1	NA	NA	86.92	87.44	**88.47**
SL 2	NA	NA	81.35	84.26	**86.78**
SL 3	NA	NA	70.95	76.81	**83.86**
SL 4	NA	NA	66.57	73.38	**81.96**
SL 5	NA	NA	59.36	66.51	**76.03**
Avg	59.00	67.00	73.03	77.68	**83.42**

Table 23. Results with Speckle noise (SL = Severity Level)

Accuracy results of our methods, TTCP and TTCP++, as well as the baseline methods with Gaussian, Impulse, Shot and Speckle noise on CIFAR10-C dataset using WRN 40-2 and WRN 28-10 architectures as student (backbone) and teacher networks respectively. Only average scores are provided for natural accuracy and JT as reported in [11]. These results show significant improvement of our method over previous methods. JT = Joint Training [11], TTT = Test time training [28]

Method	Nat.	JT[11]	TTT[28]	SSDN[1]	TTCP	TTCP++
Clean	99.03	NA	99.57	NA	**99.78**	99.53

Table 24. Results on clean MNIST test data

Method	Nat.	JT[11]	TTT[28]	SSDN[1]	TTCP	TTCP++
Clean	70.02	NA	72.11	NA	**72.65**	71.73

Table 25. Results on clean TinyImageNet test data

Accuracy comparison on clean MNIST and TinyImageNet test data using LeNet and ResNet-26 architectures respectively. These results show that, apart from improved robustness against unseen noise, our method doesn't sacrifice on clean data accuracy too.

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	78.73	83.18	80.00	81.94	**83.27**
Sev. Level 2	68.71	78.44	77.89	79.12	**81.51**
Sev. Level 3	58.65	66.88	74.49	75.76	**79.32**
Sev. Level 4	54.17	61.81	67.52	68.01	**77.76**
Sev. Level 5	49.81	56.01	56.28	56.67	**72.21**
Avg. Score	62.01	69.26	71.26	72.30	**78.91**

Table 26. Results with Gaussian noise

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	83.01	84.71	80.98	81.14	**84.79**
Sev. Level 2	75.93	81.86	78.25	79.02	**82.92**
Sev. Level 3	69.67	74.29	76.01	76.86	**81.06**
Sev. Level 4	56.01	64.09	68.56	69.21	**77.38**
Sev. Level 5	44.97	54.17	57.69	57.93	**67.14**
Avg. Score	65.92	71.82	72.29	72.83	**78.66**

Table 27. Results with Impulse noise

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	82.43	85.62	81.55	81.76	**86.29**
Sev. Level 2	77.41	82.04	79.69	79.91	**83.71**
Sev. Level 3	65.11	72.11	73.10	73.89	**79.37**
Sev. Level 4	61.02	67.38	71.92	72.45	**75.16**
Sev. Level 5	53.32	61.03	63.95	64.30	**66.29**
Avg. Score	67.86	73.64	74.02	74.46	**78.16**

Table 28. Results with Shot noise

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	83.06	83.67	79.74	80.57	**84.97**
Sev. Level 2	75.98	80.05	79.11	79.70	**83.64**
Sev. Level 3	64.76	70.02	74.32	74.98	**80.07**
Sev. Level 4	55.09	66.41	70.40	70.81	**78.36**
Sev. Level 5	50.82	59.74	64.47	64.54	**72.61**
Avg. Score	65.94	71.98	73.61	74.12	**79.93**

Table 29. Results with Speckle noise

Accuracy comparison of different parts of our method with different noises from CIFAR10-C dataset using ResNet-26 and WRN 28-10 architectures as student(backbone) and teacher networks respectively. KD = Standard model training with knowledge distillation. VQ = Accuracy measured with a standard model trained with reconstructed images from vector quantization step. TTCP = Standard model training with our proposed TTCP approach applied during inference. TTCP++ = KD + VQ + TTCP.

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	90.96	97.08	94.46	94.62	**97.18**
Sev. Level 2	87.21	93.70	93.61	93.98	**94.67**
Sev. Level 3	77.43	88.81	92.72	93.01	**93.92**
Sev. Level 4	70.47	83.97	89.35	89.79	**91.11**
Sev. Level 5	65.06	79.21	86.09	86.89	**89.26**
Avg. Score	78.22	88.55	91.24	91.65	**93.22**

Table 30. Results with Gaussian noise

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	89.38	96.61	95.83	96.03	**96.65**
Sev. Level 2	86.35	92.54	93.87	94.08	**95.09**
Sev. Level 3	82.87	90.05	91.91	92.17	**93.17**
Sev. Level 4	76.29	86.72	87.45	87.82	**91.46**
Sev. Level 5	71.92	83.28	85.24	85.56	**90.52**
Avg. Score	81.36	89.84	90.86	91.13	**93.38**

Table 31. Results with Impulse noise

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	82.03	96.23	94.76	94.85	**96.71**
Sev. Level 2	80.98	95.24	94.04	94.21	**95.81**
Sev. Level 3	78.14	90.35	93.01	93.84	**94.57**
Sev. Level 4	75.41	84.70	92.02	93.07	**93.98**
Sev. Level 5	70.18	79.56	90.68	90.87	**92.69**
Avg. Score	77.34	89.21	92.90	93.36	**94.75**

Table 32. Results with Shot noise

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	87.68	96.71	94.12	94.37	**96.79**
Sev. Level 2	84.21	93.78	93.96	94.18	**94.98**
Sev. Level 3	82.03	89.09	91.89	92.06	**93.80**
Sev. Level 4	76.99	87.16	89.26	89.61	**91.89**
Sev. Level 5	70.86	81.59	86.15	86.52	**91.38**
Avg. Score	80.35	89.66	91.07	91.34	**93.76**

Table 33. Results with Speckle noise

Accuracy comparison of different parts of our method with different noises on MNIST-C dataset using LeNet and ResNet-18 architectures as student(backbone) and teacher networks respectively. KD = Standard model training with knowledge distillation. VQ = Accuracy measured with a standard model trained with reconstructed images from vector quantization step. TTCP = Standard model training with our proposed TTCP approach applied during inference. TTCP++ = KD + VQ + TTCP.

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	56.17	58.76	56.73	56.97	**60.74**
Sev. Level 2	46.34	50.66	47.90	48.39	**56.38**
Sev. Level 3	35.28	43.92	45.61	46.04	**49.71**
Sev. Level 4	26.35	33.18	40.46	40.75	**44.12**
Sev. Level 5	21.53	27.47	35.93	36.29	**41.23**
Avg. Score	37.13	42.79	45.32	45.68	**50.43**

Table 34. Results with Gaussian noise

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	58.03	59.83	58.19	58.35	**61.72**
Sev. Level 2	48.12	51.97	48.62	48.91	**57.27**
Sev. Level 3	37.86	44.78	46.98	47.35	**50.64**
Sev. Level 4	28.41	36.57	40.14	40.78	**42.35**
Sev. Level 5	24.89	29.32	36.68	36.84	**39.18**
Avg. Score	39.46	44.49	46.04	46.45	**50.23**

Table 35. Results with Impulse noise

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	58.92	61.18	59.48	59.67	**62.98**
Sev. Level 2	50.43	54.69	50.74	50.91	**58.26**
Sev. Level 3	39.76	45.88	48.15	48.23	**50.41**
Sev. Level 4	30.54	37.04	40.85	41.06	**43.68**
Sev. Level 5	25.68	30.16	37.52	37.95	**39.53**
Avg. Score	41.06	45.79	47.35	47.56	**50.97**

Table 36. Results with Shot noise

Method	KD	TTCP	VQ	VQ+KD	TTCP++
Sev. Level 1	58.53	60.05	59.16	59.72	**62.37**
Sev. Level 2	48.76	52.61	49.27	49.73	**57.84**
Sev. Level 3	39.47	44.99	47.03	47.61	**50.02**
Sev. Level 4	28.57	36.57	39.92	40.36	**43.17**
Sev. Level 5	23.68	29.68	36.58	36.74	**39.21**
Avg. Score	39.80	44.78	46.39	46.83	**50.52**

Table 37. Results with Speckle noise

Accuracy comparison of different parts of our method with different noises on TinyImageNet-C dataset using ResNet-26 and WRN 28-10 architectures as student(backbone) and teacher networks respectively. KD = Standard model training with knowledge distillation. VQ = Accuracy measured with a standard model trained with reconstructed images from vector quantization step. TTCP = Standard model training with our proposed TTCP approach applied during inference. TTCP++ = KD + VQ + TTCP.

		Baselines		OURS	
Method	Natural	JT [11]	TTT [28]	TTCP	TTCP++
Snow	79.62	80.24	81.04	82.34	**84.67**
Frost	76.32	77.46	78.78	79.83	**81.49**
Fog	85.76	85.84	86.60	87.71	**88.09**
Zoom	81.42	81.72	83.32	**84.65**	81.92
Contrast	86.14	85.78	86.38	**87.43**	86.19

Table 38. Results of CIFAR10-C. (avg. across 5 sev. levels.)

		Baselines		OURS	
Method	Natural	JT [11]	TTT [28]	TTCP	TTCP++
Snow	15.7	15.3	17.1	17.9	**18.7**
Frost	14.9	15.8	17.9	19.2	**20.1**
Fog	15.3	17.0	20.0	21.4	**22.5**
Zoom	16.2	16.0	18.5	**19.4**	16.7
Contrast	9.7	11.0	14.4	**14.9**	11.8

Table 39. Results of ImageNet-C.(only sev. level 5.)

Accuracy results of our methods, TTCP and TTCP++, as well as the baseline methods on different corruptions such as weather based (snow, frost, fog) and camera capture based (zoom, contrast) corruptions of CIFAR10-C and ImageNet-C datasets using ResNet-26 and ResNet-18 architectures respectively.

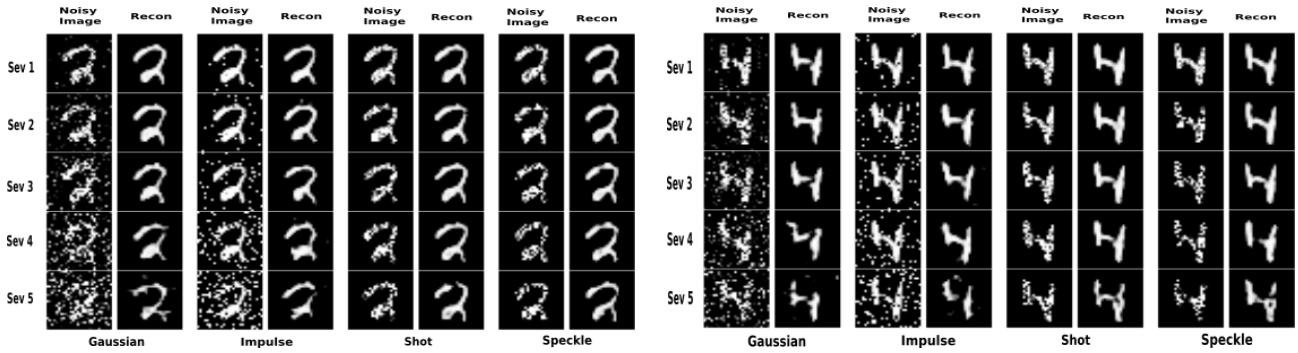

Figure 5. Visualizing effect of quantized latent on mitigating noise for two sample test images from MNIST-C dataset (Recon = reconstruction of VQ module). Note that the VQ module can address the noise at all severity levels with no prior knowledge of the noise.

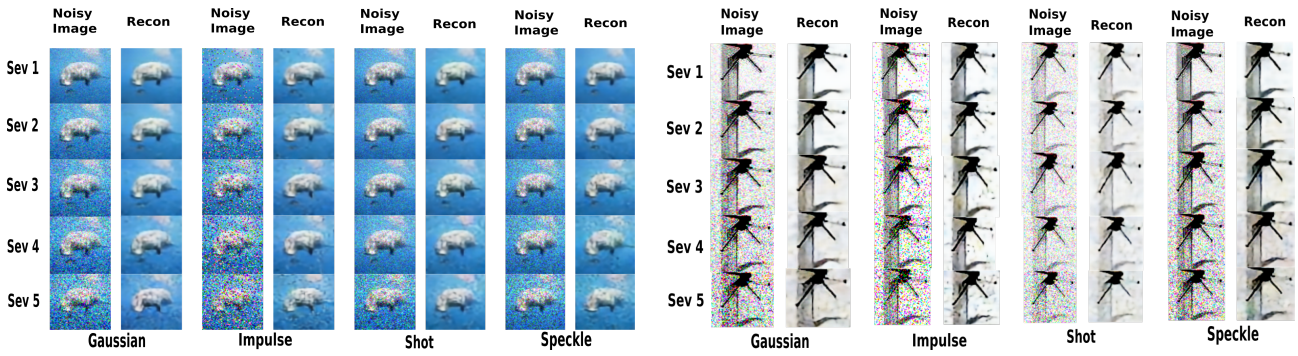

Figure 6. Visualizing effect of quantized latent on mitigating noise for two sample test images from TinyImageNet-C dataset (Recon = reconstruction of VQ module). Note that the VQ module can address the noise at all severity levels with no prior knowledge of the noise.